GEORGIA INSTITUTE OF TECHNOLOGY — OFFICE OF CONTRACT ADMINISTRATION

PROJECT ADMINISTRATION DATA SHEET

[X] ORIGINAL [ ] REVISION NO. _____

Project No. D-48-679 GTRI/~~GTRI~~ DATE 10 / 10 / 84

Project Director: John Myers School/~~Lab~~ Architecture

Sponsor: U.S. Department of the Interior - National Park Service

Type Agreement: Delivery Order No. PX-0001-4-1219 (Under BOA No. CX-0001-2-0036)

Award Period: From 9/25/84 To ~~1/25/85~~ 7-30-85 (Performance) _____ (Reports)

Sponsor Amount:	This Change	Total to Date
Estimated:	$ 9,548	$ 9,548
Funded:	$ 9,548	$ 9,548

Cost Sharing Amount: $ None Cost Sharing No: N/A

Title: "Census of Treated Historic Masonry Building - Analytical Framework"

ADMINISTRATIVE DATA OCA Contact Brian J. Lindberg x4820

1) Sponsor Technical Contact:

~~Mrs. Susan Sherwood~~

U.S. Department of the Interior
National Park Service
Preservation Assistance Division (424)
Washington, D.C. 20240
(202) 343-8149

2) Sponsor Admin/Contractual Matters:

~~Mr. John A. Thomas~~

U.S. Department of the Interior
National Park Service
Branch of Procurement (236)
Washington, D.C. 20240

Defense Priority Rating: N/A Military Security Classification: N/A

(or) Company/Industrial Proprietary: N/A

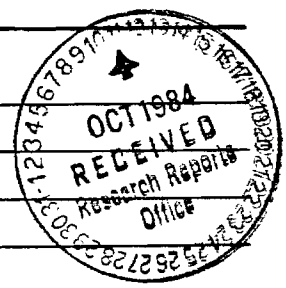
RESTRICTIONS

See Attached Gov't Supplemental Information Sheet for Additional Requirements.

Travel: Foreign travel must have prior approval -- Contact OCA in each case. Domestic travel requires sponsor approval where total will exceed greater of $500 or 125% of approved proposal budget category.

Equipment: Title vests with Sponsor if acquisition cost is $1,000 or more; However, none is proposed

COMMENTS:



COPIES TO:

I.D. #02.111.000.85.002

Project Director
Research Administrative Network
Research Property Management
Accounting
Procurement/EES Supply Services
Research Security Services
Reports Coordinator (OCA)
Research Communications (2)
GTRI
Library
Project File
Other A. Jones

FORM OCA 4.383

GEORGIA INSTITUTE OF TECHNOLOGY OFFICE OF CONTRACT ADMINISTRATION

SPONSORED PROJECT TERMINATION/CLOSEOUT SHEET

Date 1/8/86

Project No. D-48-679 School/~~Lab~~ ~~XXX~~ ARCH

Includes Subproject No.(s)

Project Director(s) John Myers GTRC ~~XXXX~~

Sponsor U. S. Department of the Interior-National Park Service

Title Census of Treated Historic Masonry Building-Analytical Framework

Effective Completion Date: 7/30/85 (Performance) 7/30/85 (Reports)

Grant/Contract Closeout Actions Remaining:

- [ ] None
- [X] Final Invoice or Final Fiscal Report
- [ ] Closing Documents
- [X] Final Report of Inventions
- [ ] Govt. Property Inventory & Related Certificate
- [ ] Classified Material Certificate
- [ ] Other

Continues Project No. Continued by Project No.

COPIES TO:

Project Director
Research Administrative Network
Research Property Management
Accounting
Procurement/GTRI Supply Services
Research Security Services
Reports Coordinator (OCA)
Legal Services

Library
GTRC
Research Communications (2)
Project File
Other Heyser
Jones
Embry

FORM OCA 69.285

June 13. 1985

Mr. Dwight Baker
Preservation Assistance Division
National Park Service (424)
P.O. Box 37127
Washington. DC 20013-7127

Dear Mr. Baker.

A summary of monthly progress and activities on PX-0001-4-1219 under contract CX-0001-2-0036 follows per our discussion.

October 1984 - Work on Task I, consisting of review and analysis of the Census program data, software options and preparation for NPS review meeting, October 17. 1984. Trip to WASO for meeting defined as Task II. 10/17/84.

November 1984 - Review of all data collected, analysis and extraction of data elements per approved plan from October review. Re-evaluation of structure based on additional work sent down by NPS on November 7, to also include a "short form" report.

December 1984 - Completion of the draft. Task III. of the program structure. Submission of that structure for approval. Preparation of recommendations for adjustments to field size and content in manual program to be more efficient in automated version.

January 1985 - Assessment of final program structure and requirements. Submission of a new recommendation to NPS on January 14 to change from Dbase III to Rbase because of potential benefits to NPS by having 1) stand-alone capability and 2) routines in PASCAL.

February 1985 - Draft program structures created in Rbase following approval. 90-day extension requested on recommendation of Anne Grimmer. System programming in progress.

March 1985 - Programming and testing of system with dummy data. Preparation of first demonstration and training for NPS on operating system.

April 1985 - Completion of menus and data entry screens. Meeting on 4/24 to demo system operation. menus. entry screens. user interface. to get NPS input prior to completion of program.

May 1985 - Completion of linkages, refinements based on 4/24 meeting and entry of live data to debug routines. Request of 90-day extension to complete debugging and refinements. Add Census buildings per Task No. 5.

June 1985 - Debugging and edit of Census buildings as specified in Task 5. Testing of completed routines. Programming of output routines. Delivery of updated Census program scheduled for 6/17/85.

This summarizes the monthly progress to date. If you would like additional details for these tasks, please let me know. As you know, monthly financial summaries and requisitions are submitted separately by our accounting office.

Sincerely,

John H. Myers
Director

JHM:dph

cc: Anne Grimmer

Baker6.13
Admin #17

SOFTWARE USER'S GUIDE

CENSUS OF TREATED HISTORIC MASONRY BUILDINGS

U.S. Department of the Interior
NATIONAL PARK SERVICE

CENSUS OF TREATED HISTORIC MASONRY BUILDINGS

SOFTWARE USER'S GUIDE

DEVELOPED FOR:

PRESERVATION ASSISTANCE DIVISION
UNITED STATES NATIONAL PARK SERVICE
DEPARTMENT OF INTERIOR
WASHINGTON, D.C. 20240

PREPARED BY:

CENTER FOR ARCHITECTURAL CONSERVATION
COLLEGE OF ARCHITECTURE
GEORGIA INSTITUTE OF TECHNOLOGY
ATLANTA, GA 30332

TABLE OF CONTENTS

INTRODUCTION

The Census of Treated Historic Masonry Buildings is a program which identifies, collects, stores and tracks information on a wide range of treatments applied to historic masonry buildings. The purpose of the Census is to provide increased levels of accurate data for monitoring the long term affects of treatments on architectural materials. The Census program is an open-ended program which collects large amounts of data on each building including multiple material study areas with multiple treatments. The manual retrieval and use of the data in the census will become increasingly difficult as the number of buildings in the program expands, however the software which is described in this user's guide will permit efficient, user-friendly entry, storage, search and retrieval of Census data.

The Census software program will allow NPS staff to do the following:

1) Add new data to the file in any of the three forms which may occur, i.e. a comprehensive Census form, a short form, and a follow-up form

2) Edit any of the forms which have been entered into the software program

3) Print complete Census forms, short forms or follow-up forms for each building

4) Print a list of all buildings and building numbers entered into the Census program

5) Delete any building form from the Census

6) Search through the data by three levels of location, three types of materials, three treatment types, and restrict the search to buildings which are Tax Act projects, National Historic Landmarks or Federally owned properties.

The Census program is totally "menu-driven", which means that the user is presented menus, or lists of options, and selects the desired option by entering a single letter which is highlighted in each menu choice. The path to specific operations is controlled by additional sub-menus and if user input is required, simple straightforward questions will prompt the user on the screen. Two conventions apply throughout the entire Census program, one is that the option desired is always selected by the entry of a single letter followed by a carriage return <CR>, and the second is that all menus have an option to exit to the previous menu and hence back out of the program to DOS.

NOTE: Software users should first become familiar with the objectives, structure and procedures of the Census of Treated Historic Masonry Buildings. This knowledge will be a prerequisite to using the Census software, since familiarity with the Census will result in

Intuitive recognition of the Census software options, structure and content. Such intuitive recognition will be necessary to effectively utilize the Census database. Prior to running the software, users should consult with the Census program manager in the Preservation Assistance Division.

The following sections outline the functions and capabilities of each of the Census options.

INSTALLING THE PROGRAM

This is a menu-driven program to allow the entry, edit, printing and searching of Census reports. The program uses the structure and selected modules from the micro-computer database program R-base, but Census routines are written in the programming language PASCAL.

The program was prepared for the Preservation Assistance Division, U.S. National Park Service, Department of Interior by the Center for Architectural Conservation, College of Architecture, Georgia Tech, Atlanta, Georgia.

I. CENSUS - INITIALLY LOADING THE PROGRAM

Step 1. Turn on computer, Boot-up to C> prompt
Step 2. At the C> prompt type: MD•CENSUS
Step 3. At the C> prompt type: CD•CENSUS
Step 4. Place the CENSUS disk in Drive A
Step 5. Type: Copy A:*.*

(PROGRAM IS LOADED ON HARD DRIVE UNDER DIRECTORY "CENSUS")

II. CENSUS - RUNNING THE PROGRAM

(If computer is on, begin with Step 2)

Step 1. Turn on computer, boot up to C> prompt
Step 2. At the C> prompt type: CD•CENSUS
Step 3. At the C> prompt type: C
Step 4. Follow menu options

NOTES:

If any unusual control characters appear on the screens, edit the file "CONFIG.SYS" on the root directory and confirm that "DEVICE = ANSI.SYS". This should clear up the screens.

This program allows any number of Masonry Study Areas per building and any number of Treatments per Study Area.

<u>It is recommended that the program and all data entered into the program be periodically backed up on floppy disks and stored safely.</u>

ADD OPTION

The "Add" option is displayed on the main menu of the Census program. This option allows the user to conduct four operations:

1) Add a Census long form
2) Add a Census follow-up form
3) Add a Census short form
4) Return to the Main Menu

The user may also exit the program from the Main Menu screen.

When the "Add" option is selected the user will be prompted by a sub-menu which displays the choices listed above. When the user makes a selection the system will ask for the two-letter abbreviation for the state in which the building is located. A unique number will then be assigned to each form and the number will consist of three parts as follows: the first two letters are of the state abbreviation, the third letter will consist of a "C" or an "S" which indicates whether the file is a long Census form or a short form, and the third part is a four-digit number which begins with 0001 and is assigned in sequence within each state. Note the following examples:

BLDG NO	BUILDING NAME	STATE
ORC0001	U.S. COURTHOUSE & POST OFFICE	OR
CAC0001	OLD SAN FRANCISCO MINT	CA

In the "Add" option, screens which are similar in appearance to the Census forms will be displayed on the CRT and the data may be entered on the screens. Upon completion of the last screen, the form will be saved. The "Add" option allows the initial entry of a form. If additional data is to be added to an existing file, it should be added via the "Edit" mode.

Adding Study Areas and Treatments:

An unlimited number of Study Areas may be entered into the program for each building. Following the entry of each Masonry Study Area, an unlimited number of Treatments may be added. Treatments are added following the Study Area to which they apply. Upon completion of Treatment entries, the user is asked if it is desired to enter another Study Area. This process will continue until the user indicates that there are no additional Study Areas or Treatments to enter. The program prompts the user to answer via simple, straightforward questions on the screen.

Entering Data:

Following the entry of data on each screen the user should type the "ESC" key. The user will then be presented with the choice of editing the screen or advancing to the next screen.

EDIT OPTION

The Edit option permits the modification of data in existing files, including the addition of new data to an existing Census file. The Edit mode is selected from the Main Menu by typing "E" followed by <CR>. The Edit Menu may be used to edit:

1) A Census long form
2) A Census short form
3) A Census follow-up form

Census long and short forms are accessed by the "Building Number", e.g. PAC0001. The user may identify the building numbers on file by printing a list of buildings in the Print mode. (A list of buildings may be displayed on the screen by executing a null query, i.e. running the Query option, but answering "N" to all questions so that all buildings will be displayed on the screen.)

Follow-up forms are accessed by the date of entry. For the convenience of the user, a list of all follow-up entry dates will be displayed on the screen after the building number has been entered.

The Edit mode is designed to allow modular access to the files via a sub-menu which displays the following organization:

LONG FORM

Building Data (Part I)
Material Data (Cont. I)
Study Area Data (Part II)
Treatment Data (Part III)
Entry Data
Follow-Up Data
Return to Main Menu

This process allows highly selective access to portions of the file so that any part of the Census file can be quickly accessed and changed. As each screen or item of information is edited, the user may save the data and go on by pressing the "ESC" key. Three choices will be presented to the user:

- Re-edit the screen on display
- Go on to the next screen in the module
- Quit and return to the sub-menu.

PRINT OPTION

The Print option allows the user to print out hard copy reports of Census files. The specific paths within this option are controlled by a sub-menu which allows the user to do the following:

1) Print a Census long form
2) Print a Census short form
3) Print all follow-up forms for a building
4) Print a list of all buildings in the Census program

The Census reports are designed to resemble the Census forms as closely as possible. The Census long form prints out in the order of entry, and each masonry study area is printed followed by all of the treatments which apply to that area. The building list prints the name, location, and building number for each building.

The selection of the report type to be printed will trigger a prompt asking for the building number of the Census report to be printed. When the number is entered there is a pause to allow the user to set up the printer, i.e. to make sure it is turned on, on-line and set to the top of the form (TOF).

(Note to Users: While there is no option to print a blank form at this time, the user can manipulate the system into printing a blank form in the following way. Create a Census file of the appropriate form to be printed, and enter one or two blank spaces in each field in the form. Once this entry has been completed, the Print option may be used to print the dummy building report which will contain all blanks, and may be used as a basic input form.)

UTILITIES OPTION

File management may be accomplished in several ways, such as printing a list of buildings via the "PRINT" option on the Main Menu. Two other important options have been built into the "Utilities Menu". They are:

1) Deleting a building from the Census
2) Deleting a follow-up form
3) Converting a "short" form to a "long" form

DELETE OPTION:

The Delete option is very straightforward; it will delete a Census long form or short form. All data in the file will be erased and the building number will be deleted from the record. Unless the building number deleted is the last number assigned within the State, the number cannot be reused. If the building deleted was the last building entered within its State, the number will be reassigned to the next building entered in that State.

OPTION TO CONVERT SHORT FORMS TO LONG FORMS

The Census Short Form is an abbreviated report which allows the program to track treatments to buildings where the full Census cannot be completed. The software program allows the entry, storage, printing, searching and deleting of "Short Forms". It is possible that at some point after a Short Form is completed and entered, the full amount of Census data will become available. Under the Utilities Menu, a Short Form may be converted to a long Census form.

The Conversion option will take the data on record in a Short Form and transfer it to the appropriate matching field in the long Census form. It also changes the building number to indicate a Census building and deletes the Short Form number. For example, a Short Form in Washington, D.C. (DCS0041) would be converted and the number in the directory or building list would become DCC0041. The Census data could then be entered to complete a long form via the "Edit" option.

QUERY OPTION

Data in the Census program may be searched for analytical and file management purposes. A query option has been designed to allow searches by the following parameters:

1) Location:

   Buildings may be searched on three levels, by:

   a) City
   b) State
   c) NPS Region

   If no location is specified, all buildings on file will be included in the search.

2) Materials:

   Buildings may be searched by up to three materials. The user simply follows the screen prompts and enters one to three materials in appropriate blocks on the screen. The buildings are searched across the materials in the "Masonry Study Area" field.

3) Treatments:

   Buildings may be searched by up to three treatments. The treatments may be combined with materials or any other search parameters.

4) Other Options:

   The search may be further restricted by three types of buildings of interest to the programs of the PAD. Buildings may be searched by whether they are:

   1) Tax Act Projects
   2) National Historic Landmarks
   3) Federally Owned Properties

Queries are executed by selecting the "Query" option and answering the questions displayed in sequence on the screen. The results of the query will be displayed on the screen, one screen at a time.

Screens may be printed by using the "Shift Key" - "Print Screen" option of DOS, i.e. hold down the shift key and touch the "PrtSc" key.

A quick technique for looking up a directory of buildings on the screen is to run a search and answer "N" to all questions. A list of all files in the program will be displayed on the screen.

CONCLUSION

The Census program is designed to provide maximum flexibility and user-friendliness in the automation of the Census of Treated Historic Masonry Buildings. Additional options and enhancements may be added to the menus as program needs require them. The microcomputer software program "Rbase" may be used to perform a variety of manipulations and reports by a user with "Rbase" experience. This allows the option of performing sophisticated data manipulation, search and analysis without additional programming. Routine system use, data entry, edit and query may be performed by personnel with minimal training and experience using the menu-driven software.

Questions and comments on the Census software program should be addressed to:

Director
Center for Architectural Conservation
College of Architecture
Georgia Institute of Technology
Atlanta, Georgia 30332
Phone: (404) 894-3390

APPENDIX A

NATIONAL REGISTER STATUS CODES
(To be used in response to Census Item #16)

CENSUS OF TREATED HISTORIC MASONRY BUILDINGS

NATIONAL REGISTER STATUS CODES

These codes are to be used to answer item #16 in the Census forms. (Use of codes 0 through 6 will result in compatibility with NPS Real Property Coding for internal buildings.)

CODE	CODE VALUE
0	Undetermined R Status
1	Entered on Register
2	Entered - Undocumented
3	Determined Eligible - Keeper of National Register
4	Determined Ineligible - Keeper of National Register
5	Determined Eligible - State Historic Preservation Officer
6	Determined Ineligible - State Historic Preservation Officer
7	Listed in an NR District
8	National Historic Landmark
9	National Monument

THE FOLLOWING APPLY TO NON-FEDERAL LISTINGS WHICH MEET NR CRITERIA

10	State Register - Individual
11	State Register - District
12	Local Register - Individual
13	Local Register - District

TECHNICAL MANUAL

CENSUS OF TREATED HISTORIC MASONRY BUILDINGS

U.S. Department of the Interior
NATIONAL PARK SERVICE

CENSUS OF TREATED HISTORIC MASONRY BUILDINGS

TECHNICAL MANUAL

DEVELOPED FOR:

PRESERVATION ASSISTANCE DIVISION
UNITED STATES NATIONAL PARK SERVICE
DEPARTMENT OF INTERIOR
WASHINGTON, D.C. 20240

PREPARED BY:

CENTER FOR ARCHITECTURAL CONSERVATION
COLLEGE OF ARCHITECTURE
GEORGIA INSTITUTE OF TECHNOLOGY
ATLANTA, GA 30332

INTRODUCTION

This Technical Manual is provided as a supplement to the Census of Treated Historic Masonry Buildings, "Software User's Guide". This volume provides the database structures and program source code necessary to evaluate and maintain the program.

The Census software is a stand-alone, menu-driven program reflecting the organization and structure of the Census. This design promotes an easy transition from the manual approach used in the early Census to this automated Census program. Users familiar with the Census objectives, structure and procedures will intuitively recognize the content and structure of the Census software.

The first part of this volume is a listing of all database "table" structures. Census data is maintained in tables compatible with the proprietary database management system (DBMS), Rbase 5000. By design, in addition to the menu-driven routines, the Census program data can be queried on an ad hoc basis, reported and edited using Rbase 5000, by users familiar with that software.

The second part of this volume is a listing of Census program source code. All program routines are listed out. The program is written in Microsoft Pascal, Version 1.0, for flexibility, speed and efficiency. The program is compiled with the Microsoft Pascal Compiler, Version 1.0. Questions concerning the program should be directed to:

Director
Center for Architectural Conservation
College of Architecture
Georgia Institute of Technology
Atlanta, Georgia 30332

SECTION A

CENSUS OF TREATED HISTORIC MASONRY BUILDINGS

DATABASE FILE STRUCTURES

Table: FOLLOW1
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	BNUMBER	TEXT	8 characters	yes
2	ENTDATE	TEXT	8 characters	yes
3	BNAME	TEXT	40 characters	
4	ONAME	TEXT	40 characters	
5	OSTREET	TEXT	30 characters	
6	OCITY	TEXT	20 characters	
7	OSTATE	TEXT	4 characters	
8	OZIP	TEXT	10 characters	
9	OPHONE	TEXT	14 characters	

Table: FOLLOW2
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	BNUMBER	TEXT	8 characters	yes
2	ENTDATE	TEXT	8 characters	yes
3	TRTDATE	TEXT	8 characters	
4	TRTLOC	TEXT	30 characters	
5	TRTTYPE	TEXT	90 characters	
6	MASTYPE	TEXT	24 characters	
7	PROD1	TEXT	60 characters	
8	PROD2	TEXT	60 characters	
9	MATCHUNG	TEXT	4 characters	
10	CHNGDES	TEXT	180 characters	
11	AREADES	TEXT	180 characters	
12	LAPSETM	TEXT	10 characters	
13	REPTRT	TEXT	4 characters	
14	REPDATE	TEXT	8 characters	
15	FUTTRT	TEXT	4 characters	
16	FUTDES	TEXT	180 characters	
17	COMMENTS	TEXT	180 characters	

Table: FOLLOW3
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	BNUMBER	TEXT	8 characters	yes
2	ENTDATE	TEXT	8 characters	yes
3	RNAME	TEXT	40 characters	
4	RORGAN	TEXT	30 characters	
5	RTITLE	TEXT	30 characters	
6	RSTREET	TEXT	30 characters	
7	RCITY	TEXT	20 characters	
8	RSTATE	TEXT	4 characters	
9	RZIP	TEXT	10 characters	
10	RPHONE	TEXT	14 characters	

Table: TREATMNT
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	BNUMBER	TEXT	8 characters	yes
2	MASTYPE	TEXT	24 characters	
3	NUMBER	INTEGER	1 value(s)	yes
4	TRTTYPE	TEXT	90 characters	
5	TNUM	INTEGER	1 value(s)	yes
6	BEGDATE	TEXT	8 characters	
7	ENDDATE	TEXT	8 characters	
8	TESTED	TEXT	4 characters	
9	TESTDES	TEXT	180 characters	
10	PROD1	TEXT	60 characters	
11	PROD2	TEXT	60 characters	
12	PROD3	TEXT	60 characters	
13	temp	TEXT	10 characters	
14	humidity	TEXT	10 characters	
15	precip	TEXT	10 characters	
16	AREATRT	TEXT	30 characters	
17	APPEAR	TEXT	180 characters	

Table: STUDY1
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	BNUMBER	TEXT	8 characters	yes
2	STUDYLOC	TEXT	30 characters	
3	MASTYPE	TEXT	24 characters	
4	NUMBER	INTEGER	1 value(s)	yes
5	MASQRRY	TEXT	24 characters	
6	MASCOUR	TEXT	24 characters	
7	MASFIN	TEXT	24 characters	
8	MASUSE	TEXT	24 characters	
9	MASCOND	TEXT	24 characters	
10	MASCOLR	TEXT	24 characters	
11	MASDENS	TEXT	10 characters	
12	MASPOROS	TEXT	10 characters	
13	MASSTREN	TEXT	10 characters	
14	ADJMAT1	TEXT	30 characters	
15	ADJMAT2	TEXT	30 characters	
16	ADJMAT3	TEXT	30 characters	
17	MORTTYPE	TEXT	10 characters	
18	MORTCOLR	TEXT	10 characters	
19	MORTSOFT	TEXT	4 characters	
20	JOINTYPE	TEXT	10 characters	
21	JOINTDEP	TEXT	16 characters	
22	MORTCOND	TEXT	10 characters	
23	MORTANAL	TEXT	4 characters	

Table: STUDY2
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	BNUMBER	TEXT	8 characters	yes
2	MASTYPE	TEXT	24 characters	
3	NUMBER	INTEGER	1 value(s)	yes
4	UNTRETCD	TEXT	60 characters	
5	COATCOND	TEXT	60 characters	
6	MOISTPRB	TEXT	60 characters	
7	TRT1	TEXT	30 characters	
8	TRTDATE1	TEXT	8 characters	
9	TRT2	TEXT	30 characters	
10	TRTDATE2	TEXT	8 characters	
11	TRT3	TEXT	30 characters	
12	TRTDATE3	TEXT	8 characters	
13	TRT4	TEXT	30 characters	
14	TRTDATE4	TEXT	8 characters	
15	SPECS	TEXT	4 characters	
16	PRECOND	TEXT	180 characters	
17	DETDES	TEXT	180 characters	

Table: IDENT1
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	BNUMBER	TEXT	8 characters	yes
2	BNAME	TEXT	40 characters	
3	ALTNAME	TEXT	40 characters	
4	BSTREET	TEXT	30 characters	
5	BCITY	TEXT	20 characters	
6	BCOUNTY	TEXT	20 characters	
7	BSTATE	TEXT	4 characters	
8	BZIP	TEXT	10 characters	
9	BUTM	TEXT	18 characters	
10	NPSREG	TEXT	4 characters	
11	FEDOWN	TEXT	4 characters	
12	NHL	TEXT	4 characters	
13	TRA	TEXT	4 characters	
14	HABSNO	TEXT	20 characters	*

* HABSNO is the field name for National Register Number which appears in the Census and Short-form screens and reports. Enough characters have been allowed to add a parenthetical note if desired, following the N.R. number.

Table: IDENT2
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	BNUMBER	TEXT	8 characters	yes
2	OSHIP	TEXT	8 characters	
3	ONAME	TEXT	40 characters	
4	OSTREET	TEXT	30 characters	
5	OCITY	TEXT	20 characters	
6	OSTATE	TEXT	4 characters	
7	OZIP	TEXT	10 characters	
8	OPHONE	TEXT	14 characters	
9	MANAGER	TEXT	40 characters	
10	MSTREET	TEXT	30 characters	
11	MCITY	TEXT	20 characters	
12	MSTATE	TEXT	4 characters	
13	MZIP	TEXT	10 characters	
14	MPHONE	TEXT	13 characters	

Table: IDENT3
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	BNUMBER	TEXT	8 characters	yes
2	OCCSTAT	TEXT	4 characters	
3	CURUSE	TEXT	70 characters	
4	ACCESS	TEXT	4 characters	
5	CONST	TEXT	20 characters	
6	FEDFUND	TEXT	30 characters	
7	NRSTAT	TEXT	4 characters	
8	SF	INTEGER	1 value(s)	
9	stories	TEXT	16 characters	

Table: IDENT4
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	BNUMBER	TEXT	8 characters	yes
2	RNAME	TEXT	40 characters	
3	RTITLE	TEXT	30 characters	
4	RORGAN	TEXT	30 characters	
5	RSTREET	TEXT	30 characters	
6	RCITY	TEXT	20 characters	
7	RSTATE	TEXT	4 characters	
8	RZIP	TEXT	10 characters	
9	RPHONE	TEXT	14 characters	
10	ENTDATE	TEXT	8 characters	

Table: MAT1
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	BNUMBER	TEXT	8 characters	yes
2	CONSTDT	TEXT	8 characters	
3	ARCH	TEXT	34 characters	
4	BUILDER	TEXT	34 characters	
5	ALTDATE1	TEXT	8 characters	
6	ALTARCH1	TEXT	34 characters	
7	ALTBLDR1	TEXT	34 characters	
8	ALTDATE2	TEXT	8 characters	
9	ALTARCH2	TEXT	34 characters	
10	ALTBLDR2	TEXT	34 characters	
11	ALTDATE3	TEXT	8 characters	
12	ALTARCH3	TEXT	34 characters	
13	ALTBLDR3	TEXT	34 characters	
14	STRUCSYS	TEXT	90 characters	
15	ROOFTYPE	TEXT	30 characters	
16	ROOFMAT	TEXT	30 characters	
17	DRAINTYP	TEXT	30 characters	
18	DRAINCON	TEXT	10 characters	

Table: MAT2
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	BNUMBER	TEXT	8 characters	yes
2	MASTYPE1	TEXT	24 characters	
3	MASQRRY1	TEXT	24 characters	
4	MASCOUR1	TEXT	24 characters	
5	MASFIN1	TEXT	24 characters	
6	MASUSE1	TEXT	24 characters	
7	MASLOC1	TEXT	24 characters	
8	MASCOND1	TEXT	24 characters	
9	MASCOLR1	TEXT	24 characters	
10	MASTYPE2	TEXT	24 characters	
11	MASQRRY2	TEXT	24 characters	
12	MASCOUR2	TEXT	24 characters	
13	MASFIN2	TEXT	24 characters	
14	MASUSE2	TEXT	24 characters	
15	MASLOC2	TEXT	24 characters	
16	MASCOND2	TEXT	24 characters	
17	MASCOLR2	TEXT	24 characters	
18	MASTYPE3	TEXT	24 characters	
19	MASQRRY3	TEXT	24 characters	
20	MASCOUR3	TEXT	24 characters	
21	MASFIN3	TEXT	24 characters	
22	MASUSE3	TEXT	24 characters	
23	MASLOC3	TEXT	24 characters	
24	MASCOND3	TEXT	24 characters	
25	MASCOLR3	TEXT	24 characters	

SECTION B

CENSUS OF TREATED HISTORIC MASONRY BUILDINGS

LISTING OF PROGRAM SOURCE CODE

```
{$include:'c.pas'}

implementation of census;

{ function and procedure declaration }

function vdoxqq;externi;

function null;externi;

procedure rbstrt; externi;

procedure rbstat ; externi;

procedure rbopen ; externi;

procedure rbfind ; externi;

procedure rbwher ; externi;

procedure rbsort; externi;

procedure rbget ; externi;

procedure rbput; externi;

procedure rbdel ; externi;

procedure rbload; externi;

procedure rbclos; externi;

procedure rbuser; externi;

procedure rbset; externi;

procedure rbwhr2; externi;

procedure intmov; externi;

procedure lnbc; externi;

procedure intcod ;externi;
procedure date; externi;

procedure strmov;externi;

procedure time;externi;

procedure scrint; externi;

procedure spchin; externi;

procedure conrdy; externi;

procedure rbform; externi;

procedure rbdraw; externi;

procedure rbedit ; externi;
procedure clscrn; externi;
procedure moveto ; externi;
procedure conout ; externi;
```

```
procedure beep; extern;


procedure bright;
        begin;
        write(esc:1,'[1m');
        end;
procedure clear_screen;
var [extern] vrBXqq, vrCXqq, vrDXqq: word;
var     ax:word;
begin
        vrBXqq := byword(07,0);       { white on black text }
        vrCXqq := byword(ch,cl);        {upper left corner of screen}
        vrDXqq := byword(dh,dl);      {lower right corner of screen}
        ax := vdoxqq(byword(6,0));
end;
procedure blink;
        begin;
        write(esc:1,'[5m');
        end;
procedure under_line;
        begin;
        write(esc:1,'[4m');
        end;
procedure normal;
        begin;
        write(esc:1,'[0m');
        end;
procedure cancel;
        begin;
        write(esc:1,'[8m');
        end;
procedure reverse;
        begin;
        write(esc:1,'[7m');
        end;
procedure scr_loc;

var [extern] vrBXqq, vrCXqq, vrDXqq: word;

var     ax:word;

begin

        vrBXqq := byword(0,0);
        vrDXqq := byword(row,col);
        ax := vdoxqq(byword(2,0));

        vrBXqq := byword(0,att);
        vrCXqq := 1;

        ax := vdoxqq(byword(9,ch));

end;

procedure cur_pos;

var [extern] vrBXqq, vrCXqq, vrDXqq: word;

var     ax:word;

begin

        vrBXqq := byword(0,0);
        vrDXqq := byword(row,col);
```

```
end;
procedure scroll_up;

var [extern] vrBXqq, vrCXqq, vrDXqq: word;

var     ax:word;
        ah, al :word;
        bh, bl :word;
        ch, cl :word;
        dh, dl :word;

begin
        ah := 6;
        al := 1;
        bh := 0;
        bl := 0;
        ch := wrd(ur);
        cl := wrd(uc);
        dh := wrd(lr);
        dl := wrd(lc);


        vrBXqq := byword(bh,bl);
        vrCXqq := byword(ch,cl);
        vrDXqq := byword(dh,dl);
        ax := vdoxqq(byword(ah,al));

end;
procedure scroll_dn;

var [extern] vrBXqq, vrCXqq, vrDXqq: word;

var     ax:word;
        ah, al :word;
        bh, bl :word;
        ch, cl :word;
        dh, dl :word;

begin
        ah := 7;
        al := 1;
        bh := 0;
        bl := 0;
        ch := wrd(ur);
        cl := wrd(uc);
        dh := wrd(lr);
        dl := wrd(lc);


        vrBXqq := byword(bh,bl);
        vrCXqq := byword(ch,cl);
        vrDXqq := byword(dh,dl);
        ax := vdoxqq(byword(ah,al));

end;
procedure rbcheck;
begin
        rbstat(rbs);
        eor[p] := rbs<>0;
        if eor[p] then
          begin
          cur_pos(24,0);
          write('Failure in ',operation:6,' operation. Error = ',rbs:3,' Path = ',p:2);
          cur_pos(0,0);
```

```
end;


procedure search;

type

  whtype = RECORD CASE INTEGER OF
              1:(STR: STRING(122));
              2:(INT: ARRAY [1..61] OF INTEGER);
       END; {end where type record}
   name10       = array [1..8] of names;


var
   answer : char;
   natt1,natt7,natt8 : integer;
   quest : packed array[1..30] of char;
   one : integer;
   i : integer;
   temp,strlen : integer;
   where1, where7,where8 : whtype;
   att11, att17, att18 : name10;
   wherao1, wherao7, wherao8 : shorts;
   wherop1, wherop7, wherop8 : name10;
   ind1, ind7, ind8 : integer;
   ct : integer;

begin
  one := 1;
  ind1 := 1;
  natt1 := 0;
  quest := '                              ';
  temp := 0;

  for i := 1 to 5 do begin
    att17[i] := '        ';
    att18[i] := '        ';
    att11[i] := '        ';
  end;
  clear_screen(23,0,23,79);
  cur_pos(23,15);
  write('Do you want to restrict the search by a location? ');
  readln(answer);
  if answer = 'Y' then begin
    cur_pos(12,1); write('CITY  ');
    cur_pos(14,1); write('STATE ');
    cur_pos(16,1); write('REGION');
    cur_pos(12,6); reverse; write(quest:15);
    cur_pos(12,6); readln(quest);
    if quest[1] = ' ' then begin
      cur_pos(12,6); normal; write(quest:15);
      cur_pos(14,7); reverse; write(quest:2);
      cur_pos(14,7); readln(quest);
      if quest[1] = ' ' then begin
        cur_pos(14,7); normal; write(quest:2);
        cur_pos(16,8); reverse; write(quest:2);
        cur_pos(16,8); readln(quest);
        if quest[1] = ' ' then begin
          cur_pos(16,8); normal; write(quest:2);
          clear_screen(23,0,23,79);
          cur_pos(23,15); blink;
          write('Invalid location; no specific location assumed');
          normal;
        end
```

```
        normal;
        attl1[1] := 'NPSREG  ';
        where1.int[1] := 4;
        for i := 1 to 4 do where1.str[i+2] := quest[i];
        natt1 := natt+1;
        wherop1[1] := 'EQ      ';
        wherao1[1] := 'AND ';
        ind1 := 4;
      end {else if region specified}
    end {if state not specified}
    else begin
      normal;
      attl1[1] := 'BSTATE  ';
      where1.int[1] := 4;
      for i := 1 to 4 do where1.str[i+2] := quest[i];
      natt1 := natt1 + 1;
      wherop1[1] := 'EQ      ';
      wherao1[1] := 'AND ';
      ind1 := 4;
    end {if state specified}
  end {if city not specied}
  else begin
    normal;
    strlen := 30;
    attl1[1] := 'BCITY   ';
    lnbc(ads quest,one,strlen,temp);
    strlen := temp;
    where1.int[ind1] := strlen;
    if ((strlen mod 2 ) <> 0) then strlen := strlen + 1;
    for i := 1 to strlen do where1.str[i+2] := quest[i];
    natt1 := natt1 + 1;
    wherop1[1] := 'CONTAINS';
    wherao1[1] := 'AND ';
    ind1 := ind1 + (strlen div 2) + 1;
  end; {if city specified}
end; {if specific location desired}

ind7 := 1;
natt7 := 0;
for i := 1 to 30 do quest[i] := ' ';
clear_screen(23,0,23,79);
cur_pos(23,15); normal;
write('Do you want to search for a specific material? ');
readln(answer);
if answer = 'Y' then begin
  cur_pos(12,25); write('1 ');
  cur_pos(14,25); write('2 ');
  cur_pos(16,25); write('3 ');
  cur_pos(12,27); reverse; write(quest:20);
  cur_pos(12,27); readln(quest); normal;
  if quest[1] <> ' ' then begin
   natt7 := natt7 + 1;
   strlen := 30;
   attl7[1] := 'MASTYPE ';
   wherop7[1] := 'CONTAINS';
   wherao7[1] := 'OR  ';
   lnbc(ads quest,one,strlen,temp);
   strlen := temp;
   where7.int[ind7] := strlen;
   if ((strlen mod 2 ) <> 0) then strlen := strlen + 1;
   for i := 1 to strlen do
     where7.str[(ind7*2)+i] := quest[i];
     ind7 := ind7 + (strlen div 2) + 1;
     cur_pos(12,27); normal; write(quest:20);
     for i := 1 to 20 do quest[i] := ' ';
```

```
    cur_pos(14,27); readln(quest);normal;
    if quest[1] <> ' ' then begin
      natt7 := natt7 + 1;
      strlen := 30;
      attl7[2] := 'MASTYPE ';
      wherop7[2] := 'CONTAINS';
      wherao7[1] := 'OR  ';
      lnbc(ads quest,one,strlen,temp);
      strlen := temp;
      where7.int[ind7] := strlen;
      if ((strlen mod 2) <> 0) then strlen := strlen + 1;
      for i := 1 to strlen do
        where7.str[(ind7*2)+i] := quest[i];
      ind7 := ind7 + (strlen div 2) + 1;
      cur_pos(14,27); normal; write(quest:20);
      for i := 1 to 20 do quest[i] := ' ';
      cur_pos(16,27); reverse; write(quest:20);
      cur_pos(16,27); readln(quest); normal;
      if quest[1] <> ' ' then begin
        natt7 := natt7 + 1;
        strlen := 30;
        attl7[3] := 'MASTYPE ';
        wherop7[3] := 'CONTAINS';
        lnbc(ads quest,one,strlen,temp);
        strlen := temp;
        where7.int[ind7] := strlen;
        if ((strlen mod 2) <> 0) then strlen := strlen + 1;
        for i := 1 to strlen do
          where7.str[(ind7*2)+i] := quest[i];
        ind7 := ind7 + (strlen div 2) + 1;
       end  {3 material types specified}
       else begin
         cur_pos(16,27); write(quest:20);
       end;
     end  {2 material types specified}
     else begin
       cur_pos(14,27); write(quest:20);
     end;
    end {1 material type specified}
    else begin
      cur_pos(12,27); write(quest:20);
    end;
end; {if material criteria desired}

ind8 := 1;
natt8 := 0;
for i := 1 to 30 do quest[i] := ' ';
clear_screen(23,0,23,79);
cur_pos(23,15);
write('Do you want to search for a specific treatment ');
readln(answer);
if answer = 'Y' then begin
  cur_pos(12,45); write('1 ');
  cur_pos(14,45); write('2 ');
  cur_pos(16,45); write('3 ');
  cur_pos(12,47); reverse; write(quest:30);
  cur_pos(12,47); readln(quest); normal;
  if quest[1] <> ' ' then begin
    strlen := 30;
    natt8 := natt8 + 1;
    attl8[1] := 'TRTTYPE ';
    wherop8[1] := 'CONTAINS';
    wherao8[1] := 'OR  ';
    lnbc(ads quest,one,strlen,temp);
    strlen := temp;
```

```
      if ((strlen mod 2) <> 0) then strlen := strlen + 1;
      for i := 1 to strlen do
        where8.str[(ind8*2)+i] := quest[i];
      ind8 := ind8 + (strlen div 2) + 1;
      cur_pos(12,47); normal; write(quest:30);
      for i := 1 to 30 do quest[i] := ' ';
      cur_pos(14,47); reverse; write(quest:30);
      cur_pos(14,47); readln(quest); normal;
      if quest[1] <> ' ' then begin
        natt8 := natt8 + 1;
        strlen := 30;
        attl8[2] := 'TRTTYPE ';
        wherop8[2] := 'CONTAINS';
        wherao8[2] := 'OR  ';
        lnbc(ads quest,one,strlen,temp);
        strlen := temp;
        where8.int[ind8] := strlen;
        if ((strlen mod 2) <> 0) then strlen := strlen + 1;
        for i := 1 to strlen do
          where8.str[(ind8*2)+i] := quest[i];
        ind8 := ind8 + (strlen div 2) + 1;
        cur_pos(14,47); normal; write(quest:30);
        for i := 1 to 30 do quest[i] := ' ';
        cur_pos(16,47); reverse; write(quest:30);
        cur_pos(16,47); readln(quest); normal;
        if quest[1] <> ' ' then begin
         natt8 := natt8 + 1;
         strlen := 30;
         attl8[3] := 'TRTTYPE ';
         wherop8[3] := 'CONTAINS';
         lnbc(ads quest,one,strlen,temp);
         strlen := temp;
         where8.int[ind8] := strlen;
         if ((strlen mod 2) <> 0) then strlen := strlen + 1;
         for i := 1 to strlen do
           where8.str[(ind8*2)+i] := quest[i];
         ind8 := ind8 + (strlen div 2) + 1;
        end   {three treatment types selected}
        else begin
          cur_pos(12,47); write(quest:30);
        end;
      end {two treatment types selected}
      else begin
        cur_pos(14,47); write(quest:30);
      end;
    end  {one treatment type selected}
    else begin
      cur_pos(16,47); write(quest:30);
    end;
  end; {treatment criteria desired}


temp := 0;
clear_screen(22,0,23,79);
cur_pos(22,1); normal;
write('Do you want to limit the search to: ');
cur_pos(23,15);
write('Federally Owned Buildings (Y/N)  ');
readln(answer);
if answer = 'Y' then begin
  natt1 := natt1 + 1;
  wherao1[natt1] := 'AND ';
  attl1[natt1] := 'FEDOWN  ';
  wherop1[natt1] := 'EQ      ';
  where1.int[ind1] := 4;
```

```
      where1.str[(ind1*2)+1] := answer;

      ind1 := ind1 + (strlen div 2) + 1;
    end;
    clear_screen(23,0,23,79);
    cur_pos(23,15);
    write('National Historic Landmarks (Y/N) ');
    readln(answer);
    if answer = 'Y' then begin
      natt1 := natt1 + 1;
      wherao1[natt1] := 'AND ';
      att11[natt1] := 'NHL     ';
      wherop1[natt1] := 'EQ      ';
      where1.int[ind1] := 4;
      for i := 2 to 4 do where1.str[(ind1*2)+i] := ' ';
        where1.str[(ind1*2)+1] := answer;
      ind1 := ind1 + (strlen div 2) + 1;
    end;
    clear_screen(23,0,23,79);
    cur_pos(23,15);
    write('Tax Act Projects    (Y/N) ');
    readln(answer);
    if answer = 'Y' then begin
      natt1 := natt1 + 1;
      wherao1[natt1] := 'AND ';
      att11[natt1] := 'TRA     ';
      wherop1[natt1] := 'EQ      ';
      where1.int[ind1] := 4;
      for i := 2 to 4 do where1.str[(ind1*2)+i] := ' ';
        where1.str[(ind1*2)+1] := answer;
      ind1 := ind1 + (strlen div 2) + 1;
    end;

  if natt7 > 0 then begin
         wherao7[natt7] := 'AND ';
         natt7 := natt7 + 1;
         where7.int[ind7] := 8;
         att17[natt7] := 'BNUMBER ';
         wherop7[natt7] := 'EQ      ';
  end;
  if natt8 > 0 then begin
         wherao8[natt8] := 'AND ';
         natt8 := natt8 + 1;
         where8.int[ind8] := 8;
         att18[natt8] := 'BNUMBER ';
         wherop8[natt8] := 'EQ      ';
  end;


   clear_screen(10,0,23,79);
   cur_pos(10,0);
   bright;
   write(' BLDG NO  BUILDING NAME                          CITY                STATE');
   normal;
   ct := 0;

   rbfind(path[1],relname[1]);
   if natt1 <> 0 then begin
    rbwher(path[1],att11[1],wherop1[1],ads where1,wherao1[1],natt1);
    rbstat(rbs);
   end
   else rbs := 0;
    if rbs <> -1 then begin
     rbget(path[1],ads row1);
     rbstat(rbs);
```

```
while not eor[1] do begin
  if natt7 > 0 then begin   {criteria specified for materials}
    for i := 1 to 8 do where7.str[(ind7*2) + i] := row1.bnumber[i];
    rbfind(path[2],relname[8]);
    rbwher(path[2],att17[1],wherop7[1],ads where7,wherao7[1],natt7);
    rbstat(rbs);
    if rbs <> -1 then begin
      if natt8 > 0 then begin    {criteria specied for treatment}
       for i := 1 to 8 do where8.str[(ind8*2) + i] := row1.bnumber[i];
       rbfind(path[3],relname[7]);
       rbwher(path[3],att18[1],wherop8[1],ads where8,wherao8[1],natt8);
       rbstat(rbs);
       if rbs <> -1 then begin
        ct := ct + 1;
        cur_pos(ct + 10,1);
        with row1 do
          write(bnumber:8,' ',bname:40,' ',bcity:20,' ',bstate:2);
       end; {if rows found}
     end {if material specified}
     else begin
       if ct = 12 then
         scroll_up(11,1,23,79)
       else
         ct := ct + 1;
       cur_pos(ct + 10,1);
       with row1 do
         write(bnumber:8,' ',bname:40,' ',bcity:20,' ',bstate:2);
    end; {if material not specified}
   end; {if treatment criteria found}
 end {if treatment specified}
 else begin
  if natt8 > 0 then begin
    for i := 1 to 8 do where8.str[(ind8*2) + i] := row1.bnumber[i];
    rbfind(path[3],relname[7]);
    rbwher(path[3],att18[1],wherop8[1],ads where8,wherao8[1],natt8);
    rbstat(rbs);
    if rbs <> -1 then begin
       if ct = 12 then
         scroll_up(11,1,23,79)
       else
         ct := ct + 1;
      cur_pos(ct + 10,1);
      with row1 do
        write(bnumber:8,' ',bname:40,' ',bcity:20,' ',bstate:2);
    end; {if rows found}
  end
  else begin
       if ct = 12 then
         scroll_up(11,1,23,79)
       else
         ct := ct + 1;
    cur_pos(ct + 10, 1);
    with row1 do
      write(bnumber:8,' ',bname:40,' ',bcity:20,' ',bstate:2);
  end;
 end; {else treatment not specified}
  rbget(path[1],ads row1);
  rbstat(rbs);
  eor[1] := rbs <> 0;
end; {while records for row1}
if ct = 0 then begin
 cur_pos(15,5);
 bright; write('No Buildings Found for Specified Criteria');
end;
cur_pos(23,1); normal;
```

```
   end {if rows found for location}
   else begin
     cur_pos(15,5);
     bright; write('No Buildings Found for Specified Criteria');
     cur_pos(23,1); normal;
     write('Enter any key to continue');
  end;
spchin(ch);
end; {procedure}


procedure blist;

begin
     attname[1] := 'BNUMBER ';
     natt_sort := 1;
     dirlist[1] := 0;
brighticur_pos(22,20);write('Prepare Printer for Building List Report');
cur_pos(23,25);write('Press any key to continue ');
spchin(ch);
        assign(f,'lpt1:');
        rewrite(f);
        write(f,compress);
        rbfind(path[2],relname[1]);
        rbcheck('rbfind  ',2);
        rbsort(path[2],attname[1],natt_sort,dirlist[1]);
        rbcheck('rbsort  ',2);
        writeln(f);
        writeln(f);
        writeln(f);
        writeln(f);
        writeln(f,' ':42,'CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
        writeln(f);
        writeln(f,' ':39,'Programmed for the Preservation Assistance Division');
        writeln(f,' ':51,'U.S. National Park Service');
        writeln(f);
        writeln(f);
        writeln(f,' ':52,'LIST OF TREATED BUILDINGS');
        writeln(f);
        writeln(f);
        writeln(f,' ':15,'Building');
        writeln(f,' ':10,'#     Number              Building Name                              Building Address',
'         Building City, State');
        writeln(f,' ':8,'-----  --------  ----------------------------------------  ------------------------------',
'  ----------------------');
         w := 0;
         repeat
            with row1 do begin
               w := w + 1;
            rbget(path[2],ads row1);
            rbstat(rbs);
            if rbs <> -1 then begin
            writeln(f,' ':8,w:4,' ':3,bnumber:8,' ':3,bname:40,' ':3,bstreet:30,' ':3,bcity:20,'  ',bstate:2);
                end;
              end;
         until rbs = -1;
         writeln(f,form_feed);
         writeln(f,nocom);
         close(f);
end;


end.
```

```
procedure open_db;
begin
        dbname := 'C:CENSUS ';
        rbstrt;
        rbopen (dbname);
end;

procedure close_db;
begin
        rbclos;
end;



procedure b_prompt;

var     b       : packed array [1..4] of char;
        b2      : packed array [1..4] of char;
        b_temp  : lstring(4);
        state_temp : packed array [1..2] of char;
        test    : boolean;

label   cont;

begin
        attlist[1] := 'BNUMBER ';
        attname[1] := 'BNUMBER ';
        wherop[1] := 'CONTAINS';
        wherao[1] := 'AND ';
        natt := 1;
        natt_sort := 1;
        dirlist[1] := 0;
        cur_pos(22,14);
        writeln('Enter the TWO LETTER State Code of the building surveyed: ');
        cur_pos(22,71);readln(state_temp);
        u := 0;
        z := 0;
{       WRITELN('*',STATE_TEMP,'*');}
        where.int[1] := 2;
          for w := 1 to 2 do begin where.str[wt2] := state_temp[w];{write('*',where.str[wt2],'*');} end;
           rbfind(path[1],relname[1]);
           rbcheck('rbfind  ',1);
           rbwher(path[1],attlist[1],wherop[1],ads where,wherao[1],natt);
           rbstat(rbs);
            if rbs = -1 then begin
              b2[1] := '0';
              b2[2] := '0';
              b2[3] := '0';
              b2[4] := '0';
              goto cont;
            end;
           rbsort(path[1],attname[1],natt_sort,dirlist[1]);
                repeat
                rbget(path[1],ads row1);
                rbstat(rbs);
{writeln(row1.bnumber);}
                until rbs = -1;
        b2[1] := row1.bnumber[4];
        b2[2] := row1.bnumber[5];
        b2[3] := row1.bnumber[6];
        b2[4] := row1.bnumber[7];
cont:
        copylst(b2,b_temp);
```

```
{writeln(u,b_temp,test); }
        u := u + 1;
        test := encode(b_temp,u:4);
{writeln(u,b_temp,test); }
     v := 1;
        repeat
           if b_temp[v] = ' ' then b_temp[v] := '0';
           v := v + 1;
        until v = 4;
        bnum[1] := state_temp[1];
        bnum[2] := state_temp[2];
        case fmen1 of
           'S','s' : bnum[3] := 'S';
           otherwise
           bnum[3] := 'C';
        end;
        bnum[4] := b_temp[1];
        bnum[5] := b_temp[2];
        bnum[6] := b_temp[3];
        bnum[7] := b_temp[4];
        bnum[8] := ' ';
        writeln(bnum:8);
        where.int[1] := 8;
          for w := 1 to 8 do where.str[w+2] := bnum[w];

end;
procedure entdate_prompt;

var     ent     : packed array [1..8] of char;

begin
        clear_screen(10,0,24,79);
        bright;cur_pos(9,35);write('ENTRY DATES');normal;
        rbfind(path[4],relname[6]);
        rbwher(path[4],attlist[1],wherop[1],ads where,wherao[1],natt);
        rbstat(rbs);
        if rbs <> -1 then begin
        n := 0;
        repeat
           rbget(path[4],ads row6);
           rbstat(rbs1);
           if rbs1 <> -1 then begin
                n := n + 1;
                cur_pos(n+10,35);write(row6.entdate:8);
           end;
        until rbs1 = -1;
        cur_pos(23,14);
        z := 0;
        writeln('Enter the Follow-up Entry Date  (i.e. 04/12/85): ');
        cur_pos(23,62);readln(ent);
        u := 0;
        z := 0;
        rbs1 := 0;
        where.int[6] := 8;
          for w := 1 to 8 do where.str[w+12] := ent[w];
     end else begin
        cur_pos(18,25);write('NO FOLLOW_UP REPORTS FOR THIS BUILDING');
        rbs1 := -1;
     end;
end;

procedure show_bnum;

begin
        bright;
```

```
        write(bnum);
        normal;
end;

procedure show_mas;
begin
        bright;
        cur_pos(24,60);
if fmen = 'E' or else fmen = 'e' then
        write('MAS: ',row8.number:2,'          ')
else write('MAS: ',mas:2,'          ');
        normal;
end;

procedure show_treat;
begin
        bright;
        cur_pos(24,68);
if fmen = 'E' or else fmen = 'e' then
        write('TREAT: ',row7.tnum:2)
else write('TREAT: ',treat:2);
        normal;
end;


procedure load_sident1;
label exit;

begin
row1.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
   rbput(path[5],ads row1)
else begin
rbfind(path[5],relname[1]);
rbcheck('rbfind 1',5);
rbload(path[5],ads row1);
               rbstat(rbs);
               if rbs <> 0 then
                  begin
                  clscrn;
                  cur_pos(24,0);bright;
                  write('Error on load, status=',rbs);
                  normal;
                  goto exit;
                  end;
               edfunct := 0;
end;
exit:
end;
procedure load_sident2;

label exit;

begin
row2.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
   rbput(path[2],ads row2)
else begin
rbfind(path[2],relname[2]);
rbcheck('rbfind 2',2);
rbload(path[2],ads row2);
               rbstat(rbs);
               if rbs <> 0 then
                  begin
                  clscrn;
```

```
                write('Error on load, status=',rbs);
                normal;
                goto exit;
                end;
             edfunct := 0;
end;
exit:
end;
procedure load_sident3;

label exit;

begin
row3.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
   rbput(path[2],ads row3)
else begin
rbfind(path[2],relname[3]);
rbcheck('rbfind 3',2);
rbload(path[2],ads row3);
             rbstat(rbs);
             if rbs <> 0 then
                begin
                clscrn;
                cur_pos(24,0);bright;
                write('Error on load, status=',rbs);
                normal;
                goto exit;
                end;
             edfunct := 0;
end;
exit:
end;
procedure load_smat1;

label exit;

begin
row9.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
   rbput(path[2],ads row9)
else begin
rbfind(path[2],relname[9]);
rbcheck('rbfind 4',2);
rbload(path[2],ads row9);
             rbstat(rbs);
             if rbs <> 0 then
                begin
                clscrn;
                cur_pos(24,0);bright;
                write('Error on load, status=',rbs);
                normal;
                goto exit;
                end;
             edfunct := 0;
end;
exit:
end;
procedure load_smat2;

label exit;

begin
row10.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
```

```
else begin
rbfind(path[2],relname[10]);
rbcheck('rbfind 5',2);
rbload(path[2],ads row10);
                rbstat(rbs);
                if rbs <> 0 then
                   begin
                   clscrn;
                   cur_pos(24,0);bright;
                   write('Error on load, status=',rbs);
                   normal;
                   goto exit;
                   end;
                edfunct := 0;
end;
exit:
end;


procedure load_sstudy1;

label exit;

begin
row8.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
   rbput(path[2],ads row8)
else begin
row8.number := mas;
rbfind(path[2],relname[8]);
rbcheck('rbfind 6',2);
rbload(path[2],ads row8);
                rbstat(rbs);
                if rbs <> 0 then
                   begin
                   clscrn;
                   cur_pos(24,0);bright;
                   write('Error on load, status=',rbs);
                   normal;
                   goto exit;
                   end;
                edfunct := 0;
end;
exit:
end;
procedure load_sstudy2;

label exit;

begin
if fmen = 'E' or else fmen = 'e' then
   rbput(path[3],ads row11)
else begin
row11.number := mas;
row11.bnumber := bnum;
row11.mastype := row8.mastype;
rbfind(path[3],relname[11]);
rbcheck('rbfind 7',3);
rbload(path[3],ads row11);
                rbstat(rbs);
                if rbs <> 0 then
                   begin
                   clscrn;
                   cur_pos(24,0);bright;
                   write('Error on load, status=',rbs);
```

```
                goto exit;
                end;
             edfunct := 0;
end;
exit:
end;
procedure load_streat1;

label exit;

begin
row7.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
   rbput(path[4],ads row7)
else begin
row7.tnum := treat;
row7.number := row11.number;
row7.mastype := row8.mastype;
rbfind(path[4],relname[7]);
rbcheck('rbfind 8',4);
rbload(path[4],ads row7);
             rbstat(rbs);
             if rbs <> 0 then
                begin
                clscrn;
                cur_pos(24,0);bright;
                write('Error on load, status=',rbs);
                normal;
                goto exit;
                end;
             edfunct := 0;
end;
exit:
end;
procedure load_sident4;

label exit;

begin
row4.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
   rbput(path[2],ads row4)
else begin
rbfind(path[2],relname[4]);
rbcheck('rbfind 9',2);
rbload(path[2],ads row4);
             rbstat(rbs);
             if rbs <> 0 then
                begin
                clscrn;
                cur_pos(24,0);bright;
                write('Error on load, status=',rbs);
                normal;
                goto exit;
                end;
             edfunct := 0;
end;
exit:
end;
procedure senter_edit(formname:names; route:integer);

var     count : integer;

label    exit;

begin
```

```
count := 1;
rbform(path[1],formname,ads formbuf);
rbstat(rbs);
if rbs <> 0 then
   begin
   cur_pos(24,0);bright;
   writeln('Error on form initialize, status=',rbs);
   normal;
   goto exit;
   end;

if fmen = 'E' or else fmen = 'e' then begin
  case route of
       1 :   begin rbget(path[5],ads row1); rbstat(rbs1);end;
       2 :   begin rbget(path[2],ads row2); rbstat(rbs1);end;
       3 :   begin rbget(path[2],ads row3); rbstat(rbs1);end;
       4 :   begin rbget(path[2],ads row9); rbstat(rbs1);end;
       6 :   begin rbget(path[2],ads row8); rbstat(rbs1);end;
       8 :   begin rbget(path[4],ads row7); rbstat(rbs1);end;
       9 :   begin rbget(path[2],ads row4); rbstat(rbs1);end;
  end;
end;

if rbs1 = -1 then goto exit;
                                    { loop, reading new rows with  }
while true do                       { a form and until quit.       }
   begin

z := 0;
repeat
  case route of
       1 :   rbdraw(path[1],ads formbuf, ads row1, edfunct);
       2 :   rbdraw(path[1],ads formbuf, ads row2, edfunct);
       3 :   rbdraw(path[1],ads formbuf, ads row3, edfunct);
       4 :   rbdraw(path[1],ads formbuf, ads row9, edfunct);
       6 :   rbdraw(path[1],ads formbuf, ads row8, edfunct);
       8 :   rbdraw(path[1],ads formbuf, ads row7, edfunct);
       9 :   rbdraw(path[1],ads formbuf, ads row4, edfunct);
  end;
   bright;                           { write escape prompt }
   cur_pos(24,0);                    { on the bottom line. }
   write('Press [ESC] when done                                        ');
   normal;
   count := 0;
   show_bnum;
   if mas <> 0 then show_mas;
   if treat <> 0 then show_treat;
  case route of
       1 :   rbedit(path[1],ads formbuf, ads row1, count,ch);
       2 :   rbedit(path[1],ads formbuf, ads row2, count,ch);
       3 :   rbedit(path[1],ads formbuf, ads row3, count,ch);
       4 :   rbedit(path[1],ads formbuf, ads row9, count,ch);
       6 :   rbedit(path[1],ads formbuf, ads row8, count,ch);
       8 :   rbedit(path[1],ads formbuf, ads row7, count,ch);
       9 :   rbedit(path[1],ads formbuf, ads row4, count,ch);
  end;
   rbstat(rbs);
   dataerr := rbs <> 0;

   if dataerr then begin
      if rbs = 34 then
         begin
         cur_pos(24,0);
         bright;
```

```
        edfunct := 1;
        cur_pos(1,0);
        end
       else
        begin
        clscrn;
        cur_pos(24,0);bright;
        writeln('Error on data entry, status=',rbs);
        normal;
        goto exit;
        end;
   end else z := 1;
until z = 1;
while true do
begin
      cur_pos(24,0);bright;
      write('N(ext), E(dit again), Q(uit)?                              ');
      normal;
      cur_pos(24,29);
      spchin(ch);

      case chr(ch) of
      'q','Q':
            begin
            clscrn;
            rbclos;
            goto exit;
            end;

      'y','Y':
            begin
            mas := 0;
            treat := 0;
            goto exit;
            end;
      'n','N':
            begin
            if not dataerr then
               begin
               case route of
                1 : load_sident1;
                2 : load_sident2;
                3 : load_sident3;
                4 : begin load_smat1; load_smat2; end;
                6 : begin load_sstudy1; load_sstudy2; end;
                8 : load_streat1;
                9 : load_sident4;
               otherwise
               cur_pos(24,0);
               write('                                          ');
               end;
                goto exit;
              end
            else
               begin
               bright;blink;
               cur_pos(24,0);
               write('Cannot load illegal data');
               cur_pos(24,25);
               spchin(ch);
               end;
            end;

      'e','E':
            begin
```

```
            break;
            end;
          otherwise
            beep;
          end;                          { case }
      end;                              { while }
    end;                                { while }
  exit:
end;                                    {form}

procedure add_short;

var     route : integer;
        screen: integer;

label   exit;

begin
mas := 0;
treat := 0;
edfunct := 0;
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ      ';
wherao[1] := 'AND ';
natt := 1;

b_prompt;
senter_edit('SIDENT1 ',1);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

edfunct := 0;
senter_edit('SIDENT2 ',2);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

edfunct := 0;
senter_edit('SIDENT3 ',3);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

edfunct := 0;
senter_edit('SMAT1   ',4);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
     it := 0;
repeat
    loop_name := 'Study Area';
    it2 := 0;
    treat := 0;
    mas := mas + 1;
    edfunct := 0;
    senter_edit('SSTUDY1 ',6);
    if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
    repeat
        treat := treat + 1;
        edfunct := 0;
        senter_edit('STREAT1 ',8);
        if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
      bright;
      cur_pos(24,0);
      write('Input Another Treatment (Y/N)? ');
      normal;
      cur_pos(24,29);
      spchin(ch);
      if chr(ch) = 'n' or else chr(ch) = 'N' then it2 := 1;
    until it2 = 1;
  bright;
  cur_pos(24,0);
  write('Input Another Study Area (Y/N)? ');
```

```
  normal;
  cur_pos(24,35);
  spchin(ch);
  if chr(ch) = 'n' or else chr(ch) = 'N' then it := 1;
until it = 1;
edfunct := 0;
senter_edit('SIDENT4 ',9);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
exit:
{load_building;}
end;

procedure edit_sident;

var     route : integer;
        screen: integer;

label   exit;

begin
mas := 0;
treat := 0;
edfunct := 1;
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ      ';
wherao[1] := 'AND ';
natt := 1;

rbfind(path[5],relname[1]);
rbwher(path[5],attlist[1],wherop[1], ads where,wherao[1],natt);
senter_edit('SIDENT1 ',1);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

edfunct := 1;
rbfind(path[2],relname[2]);
rbwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
senter_edit('SIDENT2 ',2);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

edfunct := 1;
rbfind(path[2],relname[3]);
rbwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
senter_edit('SIDENT3 ',3);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

exit:
end;
procedure edit_smat;

var     route : integer;
        screen: integer;

label   exit;

begin
mas := 0;
treat := 0;
edfunct := 1;
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ      ';
wherao[1] := 'AND ';
natt := 1;

rbfind(path[2],relname[9]);
rbwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
```

```
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

exit:
end;

procedure edit_sstudy;

var     route : integer;
        screen: integer;

label   exit;

begin
mas := 0;
treat := 0;
edfunct := 1;
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ      ';
wherao[1] := 'AND ';
natt := 1;
rbs1 := 0;

rbfind(path[2],relname[8]);
rbwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
mas := mas + 1;
senter_edit('SSTUDY1 ',6);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
until rbs1 = -1;

exit:
end;

procedure edit_streat;

var     route : integer;
        screen: integer;
        mastp : string(24);

label   exit;

begin
mas := 1;
treat := 0;
edfunct := 1;
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ      ';
wherao[1] := 'AND ';
natt := 1;
rbs1 := 0;

edfunct := 1;
rbfind(path[4],relname[7]);
rbwher(path[4],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
senter_edit('STREAT1 ',8);
mastp := row7.mastype;
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
until rbs1 = -1;

exit:
end;


procedure edit_sentry;
```

```
var     route : integer;
        screen: integer;

label   exit;

begin
mas := 0;
treat := 0;
edfunct := 1;
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ      ';
wherao[1] := 'AND ';
natt := 1;
rbs1 := 0;

edfunct := 1;
rbfind(path[2],relname[4]);
rbwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
senter_edit('SIDENT4 ',9);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
until rbs1 = -1;

exit:
end;
```

```
procedure load_ident1;

label exit;

begin
row1.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
   rbput(path[5],ads row1)
else begin
rbfind(path[5],relname[1]);
rbcheck('rbfind 1',5);
rbload(path[5],ads row1);
               rbstat(rbs);
               if rbs <> 0 then
                  begin
                  clscrn;
                  cur_pos(24,0);bright;
                  write('Error on load, status=',rbs);
                  normal;
                  goto exit;
                  end;
               edfunct := 0;
end;
exit:
end;
procedure load_ident2;

label exit;

begin
row2.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
   rbput(path[2],ads row2)
else begin
rbfind(path[2],relname[2]);
rbcheck('rbfind 2',2);
rbload(path[2],ads row2);
               rbstat(rbs);
               if rbs <> 0 then
                  begin
                  clscrn;
                  cur_pos(24,0);bright;
                  write('Error on load, status=',rbs);
                  normal;
                  goto exit;
                  end;
               edfunct := 0;
end;
exit:
end;
procedure load_ident3;

label exit;

begin
row3.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
   rbput(path[2],ads row3)
else begin
rbfind(path[2],relname[3]);
rbcheck('rbfind 3',2);
rbload(path[2],ads row3);
               rbstat(rbs);
               if rbs <> 0 then
```

```
                clscrn;
                cur_pos(24,0);bright;
                write('Error on load, status=',rbs);
                normal;
                goto exit;
                end;
             edfunct := 0;
end;
exit:
end;
procedure load_mat1;

label exit;

begin
row9.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
   rbput(path[2],ads row9)
else begin
rbfind(path[2],relname[9]);
rbcheck('rbfind 4',2);
rbload(path[2],ads row9);
             rbstat(rbs);
             if rbs <> 0 then
                begin
                clscrn;
                cur_pos(24,0);bright;
                write('Error on load, status=',rbs);
                normal;
                goto exit;
                end;
             edfunct := 0;
end;
exit:
end;
procedure load_mat2;

label exit;

begin
row10.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
   rbput(path[2],ads row10)
else begin
rbfind(path[2],relname[10]);
rbcheck('rbfind 5',2);
rbload(path[2],ads row10);
             rbstat(rbs);
             if rbs <> 0 then
                begin
                clscrn;
                cur_pos(24,0);bright;
                write('Error on load, status=',rbs);
                normal;
                goto exit;
                end;
             edfunct := 0;
end;
exit:
end;


procedure load_study1;

label exit;
```

```
begin
row8.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
   rbput(path[2],ads row8)
else begin
row8.number := mas;
rbfind(path[2],relname[8]);
rbcheck('rbfind 6',2);
rbload(path[2],ads row8);
               rbstat(rbs);
               if rbs <> 0 then
                  begin
                  clscrn;
                  cur_pos(24,0);bright;
                  write('Error on load, status=',rbs);
                  normal;
                  goto exit;
                  end;
               edfunct := 0;
end;
exit:
end;
procedure load_study2;

label exit;

begin
row11.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
   rbput(path[3],ads row11)
else begin
row11.number := mas;
row11.mastype := row8.mastype;
rbfind(path[3],relname[11]);
rbcheck('rbfind 7',3);
rbload(path[3],ads row11);
               rbstat(rbs);
               if rbs <> 0 then
                  begin
                  clscrn;
                  cur_pos(24,0);bright;
                  write('Error on load, status=',rbs);
                  normal;
                  goto exit;
                  end;
               edfunct := 0;
end;
exit:
end;
procedure load_treat1;

label exit;

begin
row7.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
   rbput(path[4],ads row7)
else begin
row7.tnum := treat;
row7.number := row11.number;
row7.mastype := row8.mastype;
rbfind(path[4],relname[7]);
rbcheck('rbfind 8',4);
rbload(path[4],ads row7);
               rbstat(rbs);
```

```
                begin
                clscrn;
                cur_pos(24,0);bright;
                write('Error on load, status=',rbs);
                normal;
                goto exit;
                end;
             edfunct := 0;
end;
exit:
end;
procedure load_ident4;

label exit;

begin
row4.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
   rbput(path[2],ads row4)
else begin
rbfind(path[2],relname[4]);
rbcheck('rbfind 9',2);
rbload(path[2],ads row4);
             rbstat(rbs);
             if rbs <> 0 then
                begin
                clscrn;
                cur_pos(24,0);bright;
                write('Error on load, status=',rbs);
                normal;
                goto exit;
                end;
             edfunct := 0;
end;
exit:
end;

procedure load_follow1;

label exit;

begin
row6.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
   rbput(path[2],ads row6)
else begin
rbfind(path[2],relname[6]);
rbcheck('rbfind 1',2);
rbload(path[2],ads row6);
             rbstat(rbs);
             if rbs <> 0 then
                begin
                clscrn;
                cur_pos(24,0);bright;
                write('Error on load, status=',rbs);
                normal;
                goto exit;
                end;
             edfunct := 0;
end;
exit:
end;
procedure load_follow2;

label exit;
```

```
begin
row12.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
   rbput(path[2],ads row12)
else begin
row12.entdate := row6.entdate;
rbfind(path[2],relname[12]);
rbcheck('rbfind 1',2);
rbload(path[2],ads row12);
               rbstat(rbs);
               if rbs <> 0 then
                  begin
                  clscrn;
                  cur_pos(24,0);bright;
                  write('Error on load, status=',rbs);
                  normal;
                  goto exit;
                  end;
               edfunct := 0;
end;
exit:
end;
procedure load_follow3;

label exit;

begin
row13.bnumber := bnum;
if fmen = 'E' or else fmen = 'e' then
   rbput(path[2],ads row13)
else begin
row13.entdate := row6.entdate;
rbfind(path[2],relname[13]);
rbcheck('rbfind 1',2);
rbload(path[2],ads row13);
               rbstat(rbs);
               if rbs <> 0 then
                  begin
                  clscrn;
                  cur_pos(24,0);bright;
                  write('Error on load, status=',rbs);
                  normal;
                  goto exit;
                  end;
               edfunct := 0;
end;
exit:
end;
procedure enter_edit(formname:names; route:integer);

var     count : integer;

label    exit;

begin

count := 1;
rbform(path[1],formname,ads formbuf);
rbstat(rbs);
if rbs <> 0 then
   begin
   cur_pos(24,0);bright;
   writeln('Error on form initialize, status=',rbs);
   normal;
   goto exit;
```

```
if fmen = 'E' or else fmen = 'e' then begin
  case route of
       1 :   begin rbget(path[5],ads row1); rbstat(rbs1);end;
       2 :   begin rbget(path[2],ads row2); rbstat(rbs1);end;
       3 :   begin rbget(path[2],ads row3); rbstat(rbs1);end;
       4 :   begin rbget(path[2],ads row9); rbstat(rbs1);end;
       5 :   begin rbget(path[2],ads row10);rbstat(rbs1); end;
       6 :   begin rbget(path[2],ads row8); rbstat(rbs1);end;
       7 :   begin rbget(path[3],ads row11);rbstat(rbs1); end;
       8 :   begin rbget(path[4],ads row7); rbstat(rbs1);end;
       9 :   begin rbget(path[2],ads row4); rbstat(rbs1);end;
      10 :   begin rbget(path[2],ads row6); rbstat(rbs1);end;
      11 :   begin rbget(path[2],ads row12); rbstat(rbs1);end;
      12 :   begin rbget(path[2],ads row13); rbstat(rbs1);end;
  end;
end;

if rbs1 = -1 then goto exit;
                                    { loop, reading new rows with  }
while true do                       { a form and until quit.       }
   begin

z := 0;
repeat
  case route of
       1 :   rbdraw(path[1],ads formbuf, ads row1, edfunct);
       2 :   rbdraw(path[1],ads formbuf, ads row2, edfunct);
       3 :   rbdraw(path[1],ads formbuf, ads row3, edfunct);
       4 :   rbdraw(path[1],ads formbuf, ads row9, edfunct);
       5 :   rbdraw(path[1],ads formbuf, ads row10, edfunct);
       6 :   rbdraw(path[1],ads formbuf, ads row8, edfunct);
       7 :   rbdraw(path[1],ads formbuf, ads row11, edfunct);
       8 :   rbdraw(path[1],ads formbuf, ads row7, edfunct);
       9 :   rbdraw(path[1],ads formbuf, ads row4, edfunct);
      10 :   rbdraw(path[1],ads formbuf, ads row6, edfunct);
      11 :   rbdraw(path[1],ads formbuf, ads row12, edfunct);
      12 :   rbdraw(path[1],ads formbuf, ads row13, edfunct);
  end;
   bright;                           { write escape prompt }
   cur_pos(24,0);                    { on the bottom line. }
   write('Press [ESC] when done                                    ');
   normal;
   count := 0;
   show_bnum;
   if mas <> 0 then show_mas;
   if treat <> 0 then show_treat;
  case route of
       1 :   rbedit(path[1],ads formbuf, ads row1, count,ch);
       2 :   rbedit(path[1],ads formbuf, ads row2, count,ch);
       3 :   rbedit(path[1],ads formbuf, ads row3, count,ch);
       4 :   rbedit(path[1],ads formbuf, ads row9, count,ch);
       5 :   rbedit(path[1],ads formbuf, ads row10, count,ch);
       6 :   rbedit(path[1],ads formbuf, ads row8, count,ch);
       7 :   rbedit(path[1],ads formbuf, ads row11, count,ch);
       8 :   rbedit(path[1],ads formbuf, ads row7, count,ch);
       9 :   rbedit(path[1],ads formbuf, ads row4, count,ch);
      10 :   rbedit(path[1],ads formbuf, ads row6, count,ch);
      11 :   rbedit(path[1],ads formbuf, ads row12, count,ch);
      12 :   rbedit(path[1],ads formbuf, ads row13, count,ch);
  end;
   rbstat(rbs);
   dataerr := rbs <> 0;

   if dataerr then begin
```

```
        begin
        cur_pos(24,0);
        bright;
        write('Data error, please correct ... Press [ESC] when done');
        edfunct := 1;
        cur_pos(1,0);
        end
      else
        begin
        clscrn;
        cur_pos(24,0);bright;
        writeln('Error on data entry, status=',rbs);
        normal;
        goto exit;
        end;
   end else z := 1;
until z = 1;
while true do
begin
     cur_pos(24,0);bright;
     write('N(ext), E(dit again), Q(uit)?                                    ');
     normal;
     cur_pos(24,29);
     spchin(ch);

     case chr(ch) of
     'q','Q':
          begin
          clscrn;
          rbclos;
          goto exit;
          end;

     'y','Y':
          begin
          mas := 0;
          treat := 0;
          goto exit;
          end;
     'n','N':
          begin
          if not dataerr then
             begin
             case route of
              1 : load_ident1;
              2 : load_ident2;
              3 : load_ident3;
              4 : load_mat1;
              5 : load_mat2;
              6 : load_study1;
              7 : load_study2;
              8 : load_treat1;
              9 : load_ident4;
             10 : load_follow1;
             11 : load_follow2;
             12 : load_follow3;
             otherwise
             cur_pos(24,0);
             write('                                          ');
             end;
              goto exit;
            end
          else
             begin
             brightblink;
```

```
            write('Cannot load illegal data');
            cur_pos(24,25);
            spchin(ch);
            end;
          end;

     'e','E':
          begin
          edfunct := 1;
          break;
          end;
       otherwise
          beep;
       end;                          { case }
     end;                            { while }
   end;                              { while }
  exit:
end;                                 {form}

procedure add_census;

var     route : integer;
        screen: integer;

label   exit;

begin
mas := 0;
treat := 0;
edfunct := 0;
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ      ';
wherao[1] := 'AND ';
natt := 1;

b_prompt;
enter_edit('IDENT1  ',1);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

edfunct := 0;
enter_edit('IDENT2  ',2);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

edfunct := 0;
enter_edit('IDENT3  ',3);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

edfunct := 0;
enter_edit('MAT1    ',4);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

edfunct := 0;
enter_edit('MAT2    ',5);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
     it := 0;
repeat
    loop_name := 'Study Area';
    it2 := 0;
    treat := 0;
    mas := mas + 1;
    edfunct := 0;
    enter_edit('STUDY1  ',6);
    if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
    edfunct := 0;
    enter_edit('STUDY2  ',7);
```

```
    repeat
        treat := treat + 1;
        edfunct := 0;
        enter_edit('TREAT1  ',8);
        if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
      bright;
      cur_pos(24,0);
      write('Input Another Treatment (Y/N)? ');
      normal;
      cur_pos(24,29);
      spchin(ch);
      if chr(ch) = 'n' or else chr(ch) = 'N' then it2 := 1;
    until it2 = 1;
  bright;
  cur_pos(24,0);
  write('Input Another Study Area (Y/N)? ');
  normal;
  cur_pos(24,35);
  spchin(ch);
  if chr(ch) = 'n' or else chr(ch) = 'N' then it := 1;
until it = 1;
edfunct := 0;
enter_edit('IDENT4  ',9);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
exit:
{load_building;}
end;

procedure edit_ident;

var     route : integer;
        screen: integer;

label   exit;

begin
mas := 0;
treat := 0;
edfunct := 1;
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ      ';
wherao[1] := 'AND ';
natt := 1;

rbfind(path[5],relname[1]);
rbwher(path[5],attlist[1],wherop[1], ads where,wherao[1],natt);
enter_edit('IDENT1  ',1);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

edfunct := 1;
rbfind(path[2],relname[2]);
rbwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
enter_edit('IDENT2  ',2);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

edfunct := 1;
rbfind(path[2],relname[3]);
rbwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
enter_edit('IDENT3  ',3);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

exit:
end;
procedure edit_mat;
```

```
        screen: integer;

label   exit;

begin
mas := 0;
treat := 0;
edfunct := 1;
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ      ';
wherao[1] := 'AND ';
natt := 1;

rbfind(path[2],relname[9]);
rbwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
enter_edit('MAT1    ',4);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

edfunct := 1;
rbfind(path[2],relname[10]);
rbwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
enter_edit('MAT2    ',5);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

exit:
end;

procedure edit_study;

var     route : integer;
        screen: integer;

label   exit;

begin
mas := 0;
treat := 0;
edfunct := 1;
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ      ';
wherao[1] := 'AND ';
natt := 1;
rbs1 := 0;

rbfind(path[2],relname[8]);
rbwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
rbfind(path[3],relname[11]);
rbwher(path[3],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
mas := mas + 1;
enter_edit('STUDY1  ',6);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

edfunct := 1;
enter_edit('STUDY2  ',7);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
until rbs1 = -1;

exit:
end;

procedure edit_treat;

var     route : integer;
        screen: integer;
```

```
label   exit;

begin
mas := 1;
treat := 0;
edfunct := 1;
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ      ';
wherao[1] := 'AND ';
natt := 1;
rbs1 := 0;

edfunct := 1;
rbfind(path[4],relname[7]);
rbwher(path[4],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
treat := treat + 1;
enter_edit('TREAT1  ',8);
mastp := row7.mastype;
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
until rbs1 = -1;

exit:
end;


procedure edit_entry;

var     route : integer;
        screen: integer;

label   exit;

begin
mas := 0;
treat := 0;
edfunct := 1;
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ      ';
wherao[1] := 'AND ';
natt := 1;
rbs1 := 0;

edfunct := 1;
rbfind(path[2],relname[4]);
rbwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
enter_edit('IDENT4  ',9);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
until rbs1 = -1;

exit:
end;

procedure edit_follow;

var     route : integer;
        screen: integer;

label   exit;

begin
mas := 0;
treat := 0;
rbs1 := 0;
```

```
entdate_prompt;
edfunct := 1;
attlist[1] := 'BNUMBER ';
attlist[2] := 'ENTDATE ';
wherop[1] := 'EQ      ';
wherop[2] := 'EQ      ';
wherao[1] := 'AND ';
wherao[2] := 'AND ';
natt := 2;

if rbs1 = -1 then goto exit;
rbfind(path[2],relname[6]);
rbwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
edfunct := 1;
enter_edit('FOLLOW1 ',10);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
edfunct := 1;
rbfind(path[2],relname[12]);
rbwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
enter_edit('FOLLOW2 ',11);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
until rbs1 = -1;
edfunct := 1;
rbfind(path[2],relname[13]);
rbwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
enter_edit('FOLLOW3 ',12);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

exit:
end;

procedure b_ask[public];

var     b       : packed array [1..4] of char;
        bnum_ent : packed array [1..8] of char;

begin
        attlist[1] := 'BNUMBER ';
        attlist[2] := 'NUMBER  ';
        attname[1] := 'BNUMBER ';
        wherop[1] := 'CON     ';
        wherop[2] := 'EQ      ';
        wherao[1] := 'AND ';
        wherao[2] := 'AND ';
        natt := 1;
        natt_sort := 1;
        dirlist[1] := 0;
        repeat
        cur_pos(22,14);
        z := 0;
        writeln('Enter the Building Number to Edit (i.e. GAC0005): ');
        cur_pos(22,64);readln(bnum_ent);
        u := 0;
        z := 0;
        where.int[1] := 8;
          for w := 1 to 8 do begin where.str[w+2] := bnum_ent[w];{write('*',where.str[w+2],'*');} end;
           rbfind(path[1],relname[1]);
           rbcheck('rbfind  ',1);
           rbwher(path[1],attlist[1],wherop[1],ads where,wherao[1],natt);
           rbstat(rbs);
                if rbs = -1 then begin
                bright;blink;
                cur_pos(23,19);write('Building Not Found ... Please Reenter');
                normal;
          end else begin
```

```
            z := 1;
          end;
      until z = 1;
      rbget(path[1],ads row1);
      bnum := row1.bnumber;
if fmen = 'E' or else fmen = 'e' then begin
      case bnum_ent[3] of
         'C' : begin
            case fmen1 of
            'B','b': edit_ident ;
            'M','m': edit_mat ;
            'S','s': edit_study ;
            'T','t': edit_treat ;
            'E','e': edit_entry ;
            'F','f': edit_follow;
            end;
            end;
         'S' : begin
            case fmen1 of
            'B','b': edit_sident ;
            'M','m': edit_smat ;
            'S','s': edit_sstudy ;
            'T','t': edit_streat ;
            'E','e': edit_sentry ;
            'F','f': edit_follow ;
            end;
            end;
      end;
  end;
end;

procedure add_follow;

var     route : integer;
        screen: integer;

label   exit;

begin
mas := 0;
treat := 0;
b_ask;
edfunct := 0;
attlist[1] := 'BNUMBER ';
wherop[1] := 'EQ      ';
wherao[1] := 'AND ';
natt := 1;

enter_edit('FOLLOW1 ',10);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;

    loop_name := 'Study Area';
    it2 := 0;
    mas := 0;
    treat := 0;
repeat
        treat := treat + 1;
        edfunct := 0;
        enter_edit('FOLLOW2 ',11);
        if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
      bright;
      cur_pos(24,0);
      write('Input Another Treatment Follow-up (Y/N)? ');
      normal;
      cur_pos(24,42);
```

```
      if chr(ch) = 'n' or else chr(ch) = 'N' then it2 := 1;
    until it2 = 1;
edfunct := 0;
enter_edit('FOLLOW3 ',12);
if chr(ch) = 'q' or else chr(ch) = 'Q' then goto exit;
exit:
end;

procedure del_disp(vars disp:integer);

begin
cur_pos(24,30);write('Removing ',disp:2,' rows');
disp := 0;
end;

procedure delete_bldg;

var       count   : integer;


begin

   attlist[1] := 'BNUMBER ';
   wherop[1] := 'EQ      ';
   wherao[1] := 'AND ';
   natt := 1;

b_ask;
count := 0;
bright;
cur_pos(23,1);write('                Removing Building from database please wait       ');
normal;
rbfind(path[1],relname[1]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row1);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
until rbs = -1;
del_disp(count);
rbfind(path[1],relname[2]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row2);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
until rbs = -1;
del_disp(count);
rbfind(path[1],relname[3]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row3);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
until rbs = -1;
del_disp(count);
rbfind(path[1],relname[4]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row4);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
until rbs = -1;
del_disp(count);
rbfind(path[1],relname[6]);
```

```
repeat
  rbget(path[1],ads row6);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
until rbs = -1;
del_disp(count);
rbfind(path[1],relname[7]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row7);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
until rbs = -1;
del_disp(count);
rbfind(path[1],relname[8]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row8);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
until rbs = -1;
del_disp(count);
rbfind(path[1],relname[9]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row9);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
until rbs = -1;
del_disp(count);
rbfind(path[1],relname[10]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row10);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
until rbs = -1;
del_disp(count);
rbfind(path[1],relname[11]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row11);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
until rbs = -1;
del_disp(count);
rbfind(path[1],relname[12]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row12);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
if rbs <> -1 then rbdel(path[1]);
until rbs = -1;
del_disp(count);
rbfind(path[1],relname[13]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row13);
  rbstat(rbs);
  if rbs <> -1 then begin rbdel(path[1]); count := count + 1; end;
until rbs = -1;
del_disp(count);
rbclos;
rbopen(dbname);
```

```
procedure convert;

var     bnum_temp : packed array [1..8] of char;

begin

   attlist[1] := 'BNUMBER ';
   wherop[1] := 'EQ      ';
   wherao[1] := 'AND ';
   natt := 1;

b_ask;
bright;
cur_pos(23,1);write('              Converting Short form to Long Census format       ');
normal;
rbfind(path[1],relname[1]);
rbcheck('find    ',1);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
rbcheck('where   ',1);
repeat
  rbget(path[1],ads row1);
  rbstat(rbs);
  if rbs <> -1 then begin
     bnum_temp := row1.bnumber;
     bnum_temp[3] := 'C';
     row1.bnumber := bnum_temp;
     rbput(path[1],ads row1);
     rbcheck('put     ',1);
  end;
until rbs = -1;
rbfind(path[1],relname[2]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row2);
  rbstat(rbs);
  if rbs <> -1 then begin
     bnum_temp := row2.bnumber;
     bnum_temp[3] := 'C';
     row2.bnumber := bnum_temp;
     rbput(path[1],ads row2);
  end;
until rbs = -1;
rbfind(path[1],relname[3]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row3);
  rbstat(rbs);
  if rbs <> -1 then begin
     bnum_temp := row3.bnumber;
     bnum_temp[3] := 'C';
     row3.bnumber := bnum_temp;
     rbput(path[1],ads row3);
  end;
until rbs = -1;
rbfind(path[1],relname[4]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row4);
  rbstat(rbs);
  if rbs <> -1 then begin
     bnum_temp := row4.bnumber;
     bnum_temp[3] := 'C';
     row4.bnumber := bnum_temp;
     rbput(path[1],ads row4);
```

```
until rbs = -1;
rbfind(path[1],relname[7]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row7);
  rbstat(rbs);
  if rbs <> -1 then begin
     bnum_temp := row7.bnumber;
     bnum_temp[3] := 'C';
     row7.bnumber := bnum_temp;
     rbput(path[1],ads row7);
  end;
until rbs = -1;
rbfind(path[1],relname[6]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row6);
  rbstat(rbs);
  if rbs <> -1 then begin
     bnum_temp := row6.bnumber;
     bnum_temp[3] := 'C';
     row6.bnumber := bnum_temp;
     rbput(path[1],ads row6);
  end;
until rbs = -1;
rbfind(path[1],relname[8]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row8);
  rbstat(rbs);
  if rbs <> -1 then begin
     bnum_temp := row8.bnumber;
     bnum_temp[3] := 'C';
     row8.bnumber := bnum_temp;
     rbput(path[1],ads row8);
  end;
until rbs = -1;
rbfind(path[1],relname[9]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row9);
  rbstat(rbs);
  if rbs <> -1 then begin
     bnum_temp := row9.bnumber;
     bnum_temp[3] := 'C';
     row9.bnumber := bnum_temp;
     rbput(path[1],ads row9);
  end;
until rbs = -1;
rbfind(path[1],relname[10]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row10);
  rbstat(rbs);
  if rbs <> -1 then begin
     bnum_temp := row10.bnumber;
     bnum_temp[3] := 'C';
     row10.bnumber := bnum_temp;
     rbput(path[1],ads row10);
  end;
until rbs = -1;
rbfind(path[1],relname[11]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row11);
```

```
  if rbs <> -1 then begin
     bnum_temp := row11.bnumber;
     bnum_temp[3] := 'C';
     row11.bnumber := bnum_temp;
     rbput(path[1],ads row11);
  end;
until rbs = -1;
rbfind(path[1],relname[12]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row12);
  rbstat(rbs);
  if rbs <> -1 then begin
     bnum_temp := row12.bnumber;
     bnum_temp[3] := 'C';
     row12.bnumber := bnum_temp;
     rbput(path[1],ads row12);
  end;
until rbs = -1;
rbfind(path[1],relname[13]);
rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
repeat
  rbget(path[1],ads row13);
  rbstat(rbs);
  if rbs <> -1 then begin
     bnum_temp := row13.bnumber;
     bnum_temp[3] := 'C';
     row13.bnumber := bnum_temp;
     rbput(path[1],ads row13);
  end;
until rbs = -1;
rbclos;
rbopen(dbname);

end;
```

```
{$DEBUG-}

interface;
unit census(pointer,names,name3,name9,dollar,shortnames,char3,char7,char8,char40,char42,char6,shorts,string8,name15,
integer5,bool6,string180,string50,whertype,rel1,rel2,rel3,rel4,rel6,rel7,rel8,rel9,rel10,rel11,rel12,rel13,
compress,form_feed,nocom,esc,rbs,rbs1,rbs2,owner,dbname,formname,status,formbuf,line,col,dataerr,blank,edfunct,n2,n1,
relname,attname,wherop,wherao,attlist,dirlist,natt,where,path,eor,row1,row2,row3,row4,row6,row7,row8,row9,row10,
row11,row12,row13,i,j,k,l,m,n,z,y,v,u,w,w1,p,chxx,ch,natt_sort,buildnum,f_num,mas,treat,it,it2,loop_name,fmen,fmen1,
fmen2,uc,ur,lc,lr,menu_choice,cls,operation,cll,choice1,choice2,choice,page_no,margin,todays_date,title,time_now,cl,
f,vdoxqq,null,rbstrt,rbstat,rbopen,rbfind,rbwher,rbsort,rbget,rbput,rbdel,rbload,rbclos,rbuser,rbset,rbwhr2,intmov,
lnbc,intcod,date,strmov,time,scrint,spchin,conrdy,rbform,rbdraw,rbedit,clscrn,moveto,conout,bline,beep,bright,clear_screen,
blink,under_line,normal,cancel,reverse,scr_loc,cur_pos,scroll_up,scroll_dn,rbcheck,search,blist,bnum);



TYPE
pointer = ADS OF WORD;
names = PACKED ARRAY [1..8] OF CHAR;
name3 = packed array [1..3] of char;
name9 = PACKED ARRAY [1..9] OF CHAR;
dollar = ARRAY [1..4] OF INTEGER;
shortnames = PACKED ARRAY [1..4] OF CHAR;
char3 = PACKED ARRAY [1..3] OF CHAR;
char7  = PACKED ARRAY [1..7] OF CHAR;
char8 = packed array [1..8] of char;
char40 = PACKED ARRAY [1..40] OF CHAR;
char42 = PACKED ARRAY [1..42] OF CHAR;
char6 = packed array [1..6] of char;
shorts = array [1..4] of shortnames;
string8 = string(8);
name15 = ARRAY [1..13] OF NAMES;
integer5 = ARRAY [1..5] OF INTEGER;
bool6 = ARRAY [1..6] OF BOOLEAN;
string180 = packed array [1..180] of char;
string50 = packed array [1..50] of char;


whertype = RECORD CASE INTEGER OF
                1:(STR: STRING(80));
                2:(I4: ARRAY [1..20] OF INTEGER4);
                3:(INT: ARRAY [1..40] OF INTEGER);
      END; {end where type record}


rel1 = record
    BNUMBER  : string(8);
    BNAME    : string(40);
    ALTNAME  : string(40);
    BSTREET  : string(30);
    BCITY    : string(20);
    BCOUNTY  : string(20);
    BSTATE   : string(4);
    BZIP     : string(10);
    BUTM     : STRING(18);
    NPSREG   : string(4);
    FEDOWN   : string(4);
    NHL      : string(4);
    TRA      : string(4);
    HABSNO   : string(20);
end; {ident1}
rel2 = record
    BNUMBER  : string(8);
    OSHIP    : string(8);
```

```
    OSTREET  : string(30);
    OCITY    : string(20);
    OSTATE   : string(4);
    OZIP     : string(10);
    OPHONE   : string(14);
    MANAGER  : string(40);
    MSTREET  : string(30);
    MCITY    : string(20);
    MSTATE   : string(4);
    MZIP     : string(10);
    MPHONE   : string(14);
end; {ident2}
rel3 = record
    BNUMBER  : string(8);
    OCCSTAT  : string(4);
    CURUSE   : string(70);
    ACCESS   : string(4);
    CNST     : string(20);
    FEDFUND  : string(30);
    NRSTAT   : string(4);
    SF       : integer4;
    STORIES  : string(16);
end; {ident3}
rel4 = record
    BNUMBER  : string(8);
    RNAME    : string(40);
    RTITLE   : string(30);
    RORGAN   : string(30);
    RSTREET  : string(30);
    RCITY    : string(20);
    RSTATE   : string(4);
    RZIP     : string(10);
    RPHONE   : string(14);
    ENTDATE  : string(8);
end; {ident4}
rel6 = record
    BNUMBER  : string(8);
    ENTDATE  : string(8);
    BNAME    : string(40);
    ONAME    : string(40);
    OSTREET  : string(30);
    OCITY    : string(20);
    OSTATE   : string(4);
    OZIP     : string(10);
    OPHONE   : string(14);
END; {follow1}
rel12 = record
    BNUMBER  : string(8);
    ENTDATE  : string(8);
    TRTDATE  : string(8);
    TRTLOC   : string(30);
    TRTTYPE  : string(90);
    MASTYPE  : string(24);
    PROD1    : string(60);
    PROD2    : string(60);
    MATCHUNG : string(4);
    CHNGDES  : string(180);
    AREADES  : string(180);
    LAPSETM  : string(10);
    REPTRT   : string(4);
    REPDATE  : string(8);
    FUTTRT   : string(4);
    FUTDES   : string(180);
    COMMENTS : string(180);
end; {follow2}
```

```
    BNUMBER  : string(8);
    ENTDATE  : string(8);
    RNAME    : string(40);
    RORGAN   : string(30);
    RTITLE   : string(30);
    RSTREET  : string(30);
    RCITY    : string(20);
    RSTATE   : string(4);
    RZIP     : string(10);
    RPHONE   : string(14);
end; {follow3}
rel7 = record
    BNUMBER  : string(8);
    MASTYPE  : string(24);
    NUMBER   : INTEGER4;
    TRTTYPE  : string(90);
    TNUM     : INTEGER4;
    BEGDATE  : string(8);
    ENDDATE  : string(8);
    TESTED   : string(4);
    TESTDES  : string(180);
    PROD1    : string(60);
    PROD2    : string(60);
    PROD3    : string(60);
    TEMP     : string(10);
    HUMIDITY : string(10);
    PRECIP   : string(10);
    AREATRT  : string(30);
    APPEAR   : string(180);
end; {treatmnt}
rel8 =  record
    BNUMBER  : string(8);
    STUDYLOC : string(30);
    MASTYPE  : string(24);
    NUMBER   : INTEGER4;
    MASQRRY  : string(24);
    MASCOUR  : string(24);
    MASFIN   : string(24);
    MASUSE   : string(24);
    MASCOND  : string(24);
    MASCOLR  : string(24);
    MASDENS  : string(10);
    MASPOROS : string(10);
    MASSTREN : string(10);
    ADJMAT1  : string(30);
    ADJMAT2  : string(30);
    ADJMAT3  : string(30);
    MORTTYPE : string(10);
    MORTCOLR : string(10);
    MORTSOFT : string(4);
    JOINTYPE : string(10);
    JOINTDEP : string(16);
    MORTCOND : string(10);
    MORTANAL : string(4);
end;{study1}
rel11 = record
    BNUMBER  : string(8);
    MASTYPE  : string(24);
    NUMBER   : INTEGER4;
    UNTRETCD : string(60);
    COATCOND : string(60);
    MOISTPR8 : string(60);
    TRT1     : string(30);
    TRTDATE1 : string(8);
    TRT2     : string(30);
    TRTDATE2 : string(8);
```

```
    TRT3     : string(30);
    TRTDATE3 : string(8);
    TRT4     : string(30);
    TRTDATE4 : string(8);
    SPECS    : string(4);
    PRECOND  : string(180);
    DETDES   : string(180);
end; {study2}
rel9 =  record
    BNUMBER  : string(8);
    CONSTDT  : string(8);
    ARCH     : string(34);
    BUILDER  : string(34);
    ALTDATE1 : string(8);
    ALTARCH1 : string(34);
    ALTBLDR1 : string(34);
    ALTDATE2 : string(8);
    ALTARCH2 : string(34);
    ALTBLDR2 : string(34);
    ALTDATE3 : string(8);
    ALTARCH3 : string(34);
    ALTBLDR3 : string(34);
    STRUCSYS : string(90);
    ROOFTYPE : string(30);
    ROOFMAT  : string(30);
    DRAINTYP : string(30);
    DRAINCON : string(10);
end; {mat1}
rel10 = record
    BNUMBER  : string(8);
    MASTYPE1 : string(24);
    MASQRRY1 : string(24);
    MASCOUR1 : string(24);
    MASFIN1  : string(24);
    MASUSE1  : string(24);
    MASLOC1  : string(24);
    MASCOND1 : string(24);
    MASCOLR1 : string(24);
    MASTYPE2 : string(24);
    MASQRRY2 : string(24);
    MASCOUR2 : string(24);
    MASFIN2  : string(24);
    MASUSE2  : string(24);
    MASLOC2  : string(24);
    MASCOND2 : string(24);
    MASCOLR2 : string(24);
    MASTYPE3 : string(24);
    MASQRRY3 : string(24);
    MASCOUR3 : string(24);
    MASFIN3  : string(24);
    MASUSE3  : string(24);
    MASLOC3  : string(24);
    MASCOND3 : string(24);
    MASCOLR3 : string(24);
end; {mat2}

CONST
        compress = chr(15);
        form_feed = chr(12);
        nocom = chr(18);
        esc = chr(27);


VAR
        rbs: INTEGER;                   {RBASE status indicator}
```

```
    rbs2: integer;

    owner: NAMES;                   {Database Owner}
    dbname: NAME9;                  {Database Name}

    formname: names;                {relation name}
    status: integer;                {Database access status}
    formbuf: array[1..50,1..30] of integer;    {form buffer}
    line, col : integer;    {screen positioning}
    dataerr: boolean;
    blank: char;                    {blank space}
    edfunct: integer;               {rbedit function specifier}
    n2,n1 : names;
    relname: NAME15;                {Relation Names}
    attname: NAME15;                {Attribute Names}
    wherop: name15;                 {RBWHER operators}
    wherao: shorts;                 {RBWHER and/or ARRAY}
    attlist: NAME15;                {Local attribute list}
    dirlist: INTEGER5;              {Direction list for RBSORT}
    natt: INTEGER;                  {Number of attributes for RBSORT}
                                    {and RBWHER}
    where: whertype;                {Type of search criteria}
    path: INTEGER5;                 {Path number}
    eor: BOOL6;                     {End-of-Relation or Error-on-Relation}
                                    {flags, depending on the context}

    row1  : rel1;                           {ident1        }
    row2  : rel2;                           {ident2        }
    row3  : rel3;                           {ident3        }
    row4  : rel4;                           {ident4        }
    row6  : rel6;                           {follow1       }
    row7  : rel7;                           {treatmnt      }
    row8  : rel8;                           {study1        }
    row9  : rel9;                           {mat1          }
    row10 : rel10;                          {mat2          }
    row11 : rel11;                          {study2        }
    row12 : rel12;                          {follow2       }
    row13 : rel13;                          {follow3       }
    i,j,k,l,m,n,z,y,v,u,w,w1,p :INTEGER;    {Scratch variables}
    chxx    : char;
    ch      : integer;
    natt_sort: integer;
    buildnum : string(8);
    bnum: names;                    {Local variable for Building number}
    f_num: string(8);
    mas : integer;
    treat : integer;
    it : integer;
    it2 : integer;
    loop_name : string(10);
    fmen: CHAR;                     {Local variable for menu choices}
    fmen1: char;
    fmen2: char;
    uc,ur,lc,lr : integer;
    menu_choice : char;
    cls : char3;
operation : string(6);
    cll : char3;
    choice1 : char;
    choice2 : char;
    choice  : char;
    page_no : integer;
    margin : integer;
    todays_date : names;
    time_now : string(8);
    title:string(75);
```

```
        cl:text;
        f:text;

{ function and procedure declaration }

function vdoxqq(areg:word):word;

function null(varname: pointer): boolean;

procedure rbstrt;

procedure rbstat (vars rbs: integer);

procedure rbopen (vars dbname: name9);

procedure rbfind (vars path: integer; vars relname: names);

procedure rbwher (vars path: integer; vars att,ops: names;
                  values: pointer; vars andor: shortnames;
                  vars nboo: integer);

procedure rbsort (vars path: integer; vars att: names;
                   vars natt, direction: integer);

procedure rbget (vars path: integer; tuple: pointer);

procedure rbput (vars path: integer; tuple: pointer);

procedure rbdel (vars path: integer);

procedure rbload (vars path: integer; tuple: pointer);

procedure rbclos;

procedure rbuser (vars pwname: names);

procedure rbset (vars set1: names; vars set2: char);

procedure rbwhr2 (vars path: integer; vars att,ops: names;
                  values: pointer; vars andor: shortnames;
                  vars nboo: integer);

procedure intmov (pstr:pointer; vars start, length: integer;
                  vars rimint: integer4);

procedure lnbc(pstr:pointer;vars start,nchar,last:integer);

procedure intcod (pstr: pointer; vars count: integer;
                  vars rimint: integer4; vars fillchar: char);
procedure date(var d: string);

procedure strmov(string1:names;pos1:integerinchar:integerivars string2:lstring;pos2:integer);

procedure time(var t: string);

procedure scrint;

procedure spchin(vars ch: integer);

procedure conrdy(vars chr,state: integer);

procedure rbform (vars path: integer; vars form: names; buff: pointer);

procedure rbdraw (vars path: integer; buff, row: pointer; vars funct: integer);
```

```
                vars item,ch: integer);
procedure clscrn;
procedure moveto (vars line, col: integer);
procedure conout (str: pointer; vars count: integer);
procedure bline (vars linen: integer);
procedure beep;
procedure bright;
procedure clear_screen(ch,cl,dh,dl:word);
procedure blink;
procedure under_line;
procedure normal;
procedure cancel;
procedure reverse;
procedure scr_loc(row,col:integer;ch,att:char);
procedure cur_pos(row,col:integer);
procedure scroll_up(ur,uc,lr,lc:integer);
procedure scroll_dn(ur,uc,lr,lc:integer);
procedure rbcheck(operation:names;p:integer);
procedure search;
procedure blist;

begin
end;
```

```
procedure util_menu;
        begin
        j := 0;
        repeat
          clear_screen(0,0,24,79);
          cur_pos(2,18);writeln('CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
          cur_pos(4,15);writeln('Programmed for the Preservation Assistance Division');
          cur_pos(5,27);writeln('U.S. National Park Service');
          bright;
          cur_pos(7,28);writeln('');
          cur_pos(8,28);writeln(' U T I L I T Y   M E N U ');
          cur_pos(9,28);writeln('');
          normal;
          cur_pos(11,28);bright;write('C');normal;write('onvert Short to Long Form');
          cur_pos(12,28);bright;write('D');normal;write('elete a Short or Long Form');
          cur_pos(13,28);bright;write('R');normal;write('eturn to Main Menu      ');
          cur_pos(14,28);bright;write(' ');normal;write('                       ');
          cur_pos(15,28);bright;write(' ');normal;write('         ');
          cur_pos(16,28);bright;write(' ');normal;write('             ');
          cur_pos(17,28);bright;write(' ');normal;write('                 ');
          cur_pos(22,29);write('Enter the option desired: ');
          repeat cur_pos(22,55); write(' '); cur_pos(22,56); readln(fmen1);
          case fmen1 of
          'C','c':  begin i := 1; convert ;j := 1; end;
          'D','d':  begin i := 1; delete_bldg ;j := 1; end;
          'R','r':  begin i := 1; j := 1; end;
{         ' ',' ':  begin i := 1; j := 0; end;
          ' ',' ':  begin i := 1; j := 0; end;
          ' ',' ':  begin i := 1; j := 0; end;
          ' ',' ':  begin i := 1; j := 1; end;
}         otherwise
                        begin
                        blink; cur_pos(23,26); write('INVALID ENTRY -- Please Re-enter');
                        normal;
                        i := 0;
                        cycle;
                        end;
          end;
          until i = 1;
        until j = 1;
end;
procedure edit_menu;
        begin
        j := 0;
        repeat
          clear_screen(0,0,24,79);
          cur_pos(2,18);writeln('CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
          cur_pos(4,15);writeln('Programmed for the Preservation Assistance Division');
          cur_pos(5,27);writeln('U.S. National Park Service');
          bright;
          cur_pos(7,30);writeln('');
          cur_pos(8,31);writeln('E D I T   M E N U ');
          cur_pos(9,30);writeln('');
          normal;
          cur_pos(11,28);bright;write('B');normal;write('uilding data (Part I)');
          cur_pos(12,28);bright;write('M');normal;write('aterial data (cont I)');
          cur_pos(13,28);bright;write('S');normal;write('tudy area data (Part II)');
          cur_pos(14,28);bright;write('T');normal;write('reatment data (Part III)');
          cur_pos(15,28);bright;write('E');normal;write('ntry data');
          cur_pos(16,28);bright;write('F');normal;write('ollow-up data');
          cur_pos(17,28);bright;write('R');normal;write('eturn to Main Menu');
          cur_pos(22,29);write('Enter the option desired: ');
          repeat cur_pos(22,55); write(' '); cur_pos(22,56); readln(fmen1);
          case fmen1 of
```

```
          'M','m':  begin i := 1; b_ask ;j := 0; end;
          'S','s':  begin i := 1; b_ask ;j := 0; end;
          'T','t':  begin i := 1; b_ask ;j := 0; end;
          'E','e':  begin i := 1; b_ask ;j := 0; end;
          'F','f':  begin i := 1; b_ask ;j := 0; end;
          'R','r':  begin i := 1; j := 1; end;
          otherwise
                        begin
                        blink; cur_pos(23,26); write('INVALID ENTRY -- Please Re-enter');
                        normal;
                        i := 0;
                        cycle;
                        end;
          end;
          until i = 1;
        until j = 1;
end;
procedure add_menu;
        begin
        j := 0;
        repeat
          clear_screen(0,0,24,79);
          cur_pos(2,18);writeln('CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
          cur_pos(4,15);writeln('Programmed for the Preservation Assistance Division');
          cur_pos(5,27);writeln('U.S. National Park Service');
          bright;
          cur_pos(7,31);writeln('');
          cur_pos(8,31);writeln('  A D D   M E N U ');
          cur_pos(9,31);writeln('');
          normal;
          cur_pos(11,28);bright;write('C');normal;write('ensus form ');
          cur_pos(12,28);bright;write('F');normal;write('ollow-up form        ');
          cur_pos(13,28);bright;write('S');normal;write('hort form             ');
          cur_pos(14,28);bright;write('R');normal;write('eturn to Main Menu');
          cur_pos(22,29);write('Enter the option desired: ');
          repeat cur_pos(22,55); write(' '); cur_pos(22,56); readln(fmen1);
          case fmen1 of
          'C','c':  begin i := 1; add_census;j := 1; end;
          'F','f':  begin i := 1; fmen1 := ' ';add_follow ;j := 1; end;
          'S','s':  begin i := 1; add_short ;j := 1; end;
          'R','r':  begin i := 1; j := 1; end;
          otherwise
                        begin
                        blink; cur_pos(23,26); write('INVALID ENTRY -- Please Re-enter');
                        normal;
                        i := 0;
                        cycle;
                        end;
          end;
          until i = 1;
        until j = 1;
end;
procedure report_menu;
begin
        j := 0;
        repeat
          clear_screen(0,0,24,79);
          cur_pos(2,18);writeln('CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
          cur_pos(4,15);writeln('Programmed for the Preservation Assistance Division');
          cur_pos(5,27);writeln('U.S. National Park Service');
          bright;
          cur_pos(7,31);writeln('');
          cur_pos(8,32);writeln('R E P O R T   M E N U');
          cur_pos(9,31);writeln('');
          normal;
          cur_pos(11,28);bright;write('C');normal;write('ensus Report');
```

```
        cur_pos(12,28);bright;write('F');normal;write('ollow-up Report     ');
        cur_pos(13,28);bright;write('B');normal;write('uilding List       ');
        cur_pos(14,28);bright;write('R');normal;write('eturn to Main Menu');
        cur_pos(22,29);write('Enter the option desired: ');
        repeat cur_pos(22,55); write(' '); cur_pos(22,56); readln(fmen1);
        case fmen1 of
        'C':  begin i := 1; comp ;j := 0; end;
        'F':  begin i := 1; print_follow; j := 0; end;
        'B':  begin i := 1; blist; j := 0; end;
        'R','r':  begin i := 1; j := 1; end;
        otherwise
                        begin
                        blink; cur_pos(23,26); write('INVALID ENTRY -- Please Re-enter');
                        normal;
                        i := 0;
                        cycle;
                        end;
        end;
        until i = 1;
      until j = 1;
end;
procedure query_menu;
      begin
      j := 0;
      repeat
        clear_screen(0,0,24,79);
        cur_pos(2,18);writeln('CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
        cur_pos(4,15);writeln('Programmed for the Preservation Assistance Division');
        cur_pos(5,27);writeln('U.S. National Park Service');
        bright;
        cur_pos(7,28);writeln('');
        cur_pos(8,29);writeln('     Q U E R Y    ');
        cur_pos(9,28);writeln('');
        normal;
      search;
      j := 1;
      until j = 1;
end;
procedure exit;
      begin
            i := 1; j := 1; k := 1
      end;

procedure main_menu;
      begin
      j := 0;
      repeat
        clear_screen(0,0,24,79);
        cur_pos(2,18);writeln('CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
        cur_pos(4,15);writeln('Programmed for the Preservation Assistance Division');
        cur_pos(5,27);writeln('U.S. National Park Service');
        bright;
        cur_pos(7,29);writeln('');
        cur_pos(8,31);writeln('M A I N   M E N U ');
        cur_pos(9,29);writeln('');
        normal;
        cur_pos(11,23);bright;write('A');normal;write('dd information to CENSUS database');
        cur_pos(12,23);bright;write('E');normal;write('dit information in CENSUS database');
        cur_pos(13,23);bright;write('P');normal;write('rint standard CENSUS reports  ');
        cur_pos(14,23);bright;write('Q');normal;write('uery the CENSUS database  ');
        cur_pos(15,23);bright;write('U');normal;write('se CENSUS database Utilities');
        cur_pos(16,23);write('e');bright;write('X');normal;write('it to DOS');
        cur_pos(22,29);write('Enter the option desired: ');
        repeat cur_pos(22,55); write(' '); cur_pos(22,56); readln(fmen);
        case fmen of
        'A','a': begin open_db; add_menu; close_db; i := 1; end;
```

```
        'E','e': begin open_db; edit_menu; close_db; i := 1; end;
        'P','p': begin open_db; report_menu; close_db; i := 1; end;
        'Q','q': begin open_db; query_menu; close_db; i := 1; end;
        'U','u': begin open_db; util_menu; close_db; i := 1;end;
        'X','x': begin open_db; exit; close_db; i := 1; k := 1;end;
        otherwise
                        begin
                        blink; cur_pos(23,26); write('INVALID ENTRY -- Please Re-enter');
                        normal;
                        i := 0;
                        cycle;
                        end;
        end;
        until i = 1;
    until k = 1;
end;
```

```
Interface;

unit crep2(b_ask,print_follow,comp);
procedure b_ask;
procedure print_follow;
procedure comp;

begin
end;

{$include:'c.pas'}

implementation of crep2;

uses census;

procedure b_ask;extern;

procedure bstring(instring:string180;vars outstr1,outstr2,outstr3,outstr4:string50);

var
  tempstring : string50;
  strlen : integer;
  i, index : integer;
  stpt : integer;
  outstr : array [1..4] of string50;


begin
  if null(ads instring) then begin
     outstr1 := '                                                  ';
     outstr2 := '                                                  ';
     outstr3 := '                                                  ';
     outstr4 := '                                                  ';
   end else begin
  for index := 1 to 4 do
  for i := 1 to 50 do outstr[index][i] := ' ';
  for i := 1 to 50 do tempstring[i] := instring[i];
    stpt := 0;
  for index := 1 to 4 do begin
    strlen := scaneq(-10,' ',tempstring,50);
    if (strlen <> -10) or (strlen <> 50) then begin
      for i := 1 to (50+strlen) do outstr[index][i] := tempstring[i];
      if strlen <> 0 then for i := (50+strlen+1) to 50 do outstr[index][i] := ' ';
    end
    else
      write('error in string search ', index:2);
    stpt := stpt + 50 + strlen;
    for i := 1 to 50 do tempstring[i] := instring[stpt+i];
  end; {for}
  outstr1 := outstr[1];
  outstr2 := outstr[2];
  outstr3 := outstr[3];
  outstr4 := outstr[4];
 end;
end;
procedure part_one_long;

var     margin  : integer;

begin
        margin := 5;
        rbfind(path[2],relname[2]);
        rbwher(path[2],attlist[1],wherop[1],ads where,whereao[1],natt);
```

```
assign(cl,'lpt1:');
rewrite(cl);
writeln(cl);
writeln(cl,' ':26,'U. S. National Park Service');
writeln(cl);
writeln(cl);
if null(ads row1.nhl) then row1.nhl := '    ';
if null(ads row1.fedown) then row1.fedown := '    ';
if null(ads row1.tra) then row1.tra := '    ';
writeln(cl,' ':margin,' ');
writeln(cl,' ':margin,'            CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
writeln(cl,' ':margin,' ');
writeln(cl,' ':margin,'                          CENSUS REPORT          ');
writeln(cl,' ':margin,' ');
writeln(cl,' ':margin,'National Historic Landmark: ',row1.nhl:3,'Federal Property: ',row1.fedown:3,'Tax Act Project: ',
          row1.tra:3);
writeln(cl,' ':margin,'-------------------------------------------------------------------------------');
writeln(cl,' ':margin,'PART I     B A C K G R O U N D   I N F O R M A T I O N ');
writeln(cl,' ':margin,'-------------------------------------------------------------------------------');
writeln(cl,' ':margin,'NAME/LOCATION');
writeln(cl,' ':margin,'1.  ',row1.bname:40,'(',row1.bnumber:7,')');
writeln(cl,' ':margin,'   Historical Name of Structure');
writeln(cl,' ':margin,'2. ',row1.altname:40);
writeln(cl,' ':margin,'   Contemporary Name of Structure (if different)');
writeln(cl,' ':margin,'3.  ',row1.bstate:2,'                        ',row1.npsreg:2);
writeln(cl,' ':margin,'   State                    NPS Region');
writeln(cl,' ':margin,'4. ',row1.bcounty:20);
writeln(cl,' ':margin,'   County');
writeln(cl,' ':margin,'5. ',row1.bcity:20);
writeln(cl,' ':margin,'   City');
writeln(cl,' ':margin,'6. ',row1.bstreet:30);
writeln(cl,' ':margin,'   Street Address (ex. 440 G Street NW)');
writeln(cl,' ':margin,'7. ',row1.bzip:10);
writeln(cl,' ':margin,'   Zip Code');
writeln(cl,' ':margin,'7a. ',row1.habsno:20,'        ',row1.butm:18);
writeln(cl,' ':margin,'   N.R. number                 UTM Coordinates');
writeln(cl,' ':margin,'OWNERSHIP/MANAGEMENT');
writeln(cl,' ':margin,'8. ',row2.oship:8);
writeln(cl,' ':margin,'    Ownership');
writeln(cl,' ':margin,'9. ',row2.oname:40,'             ', row2.ophone:14);
writeln(cl,' ':margin,'    Name of Primary Owner(s)                              Phone #'          );
writeln(cl,' ':margin,'   ',row2.ostreet:30);
writeln(cl,' ':margin,'    Street Address');
writeln(cl,' ':margin,'   ',row2.ocity:20,'        ',row2.ostate:2,'                    ',row2.ozip:10);
writeln(cl,' ':margin,'    City                     State                         Zip');
if null(ads row2.manager) then row2.manager := '                                        ';
writeln(cl,' ':margin,' 10. ',row2.manager:40,'           ',row2.mphone:14);
writeln(cl,' ':margin,'     Building Manager/Superintendent/Project Architect   Phone #');
writeln(cl,' ':margin,'     ',row2.mstreet:30);
writeln(cl,' ':margin,'     Street Address');
writeln(cl,' ':margin,'     ',row2.mcity:20,'      ',row2.mstate:2,'                    ',row2.mzip:10);
writeln(cl,' ':margin,'     City                     State                         Zip');
writeln(cl,' ':margin);
rbfind(path[2],relname[3]);
rbwher(path[2],attlist[1],wherop[1],ads where,wherao[1],natt);
rbget(path[2],ads row3);
rbfind(path[2],relname[9]);
rbwher(path[2],attlist[1],wherop[1],ads where,wherao[1],natt);
rbget(path[2],ads row9);
rbfind(path[2],relname[10]);
rbwher(path[2],attlist[1],wherop[1],ads where,wherao[1],natt);
rbget(path[2],ads row10);
writeln(cl,' ':margin,'USE/OCCUPANCY/STATUS' );
writeln(cl,' ':margin);
if null(ads row3.occstat) then row3.occstat := '    ';
```

```
        if null(ads row3.access) then row3.access := '    ';
        if null(ads row3.cnst) then row3.cnst := '                    ';
        if null(ads row3.fedfund) then row3.fedfund := '                              ';
        if null(ads row3.nrstat) then row3.nrstat := '    ';
        writeln(cl,' ':margin,'11. ',row3.occstat:4);
        writeln(cl,' ':margin,'    Percent Occupied');
        writeln(cl,' ':margin,'12. ',row3.curuse:65);
        writeln(cl,' ':margin,'    Present Use');
        writeln(cl,' ':margin,'13. ',row3.access:4);
        writeln(cl,' ':margin,'    Accessible');
        writeln(cl,' ':margin,'14. ',row3.cnst:20);
        writeln(cl,' ':margin,'    Type of Construction Work in progress or pending');
        writeln(cl,' ':margin,'15. ',row3.fedfund:30);
        writeln(cl,' ':margin,'    Explain how federal funds are involved in this project');
        writeln(cl,' ':margin,'16. ',row3.nrstat:4);
        writeln(cl,' ':margin,'    Designation/ National Historic Register');
        writeln(cl,' ':margin);
        writeln(cl,form_feed);
{   PAGE TWO   }
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl,' ':margin,'-------------------------------------------------------------------------');
        writeln(cl,' ':margin,'(cont)     B U I L D I N G   M A T E R I A L   D A T A ');
        writeln(cl,' ':margin,'-------------------------------------------------------------------------');
        writeln(cl,' ':margin,'BUILDING DESCRIPTION');
        writeln(cl,' ':margin);
        writeln(cl,' ':margin,'17. Dates of construction');
        if null(ads row9.constdt) then row9.constdt := '       ';
        if null(ads row9.arch) then row9.arch := '                                ';
        if null(ads row9.builder) then row9.builder := '                                  ';
        if null(ads row9.altdate1) then row9.altdate1 := '       ';
        if null(ads row9.altarch1) then row9.altarch1 := '                                  ';
        if null(ads row9.altbldr1) then row9.altbldr1 := '                                  ';
        if null(ads row9.altdate2) then row9.altdate2 := '       ';
        if null(ads row9.altarch2) then row9.altarch2 := '                                  ';
        if null(ads row9.altbldr2) then row9.altbldr2 := '                                  ';
        if null(ads row9.altdate3) then row9.altdate3 := '       ';
        if null(ads row9.altarch3) then row9.altarch3 := '                                  ';
        if null(ads row9.altbldr3) then row9.altbldr3 := '                                  ';
        writeln(cl,' ':margin-4,'-------------------------------------------------------------------------------');
        writeln(cl,' ':margin-4,'|  Year  |           Architect             |       Builder/contractor      |');
        writeln(cl,' ':margin-4,'|--------|---------------------------------|-------------------------------|');
        writeln(cl,' ':margin-4,'|',row9.constdt:8,'|',row9.arch:34,'|',row9.builder:33,'|');
        writeln(cl,' ':margin-4,'|--------|---------------------------------|-------------------------------|');
        writeln(cl,' ':margin-4,'|',row9.altdate1:8,'|',row9.altarch1:34,'|',row9.altbldr1:33,'|');
        writeln(cl,' ':margin-4,'|',row9.altdate2:8,'|',row9.altarch2:34,'|',row9.altbldr2:33,'|');
        writeln(cl,' ':margin-4,'|',row9.altdate3:8,'|',row9.altarch3:34,'|',row9.altbldr3:33,'|');
        writeln(cl,' ':margin-4,'|--------|---------------------------------|-------------------------------|');
        writeln(cl,' ':margin,'        ');
        if null(ads row3.sf) then row3.sf := 0;
        writeln(cl,' ':margin,'18.            ',row3.stories:16,'                    ',row3.sf:9);
        writeln(cl,' ':margin,'    Size - Height (approx. feet or stories)       Square feet (approx.)');
        writeln(cl,' ':margin,'19. ',row9.strucsys:65);
        writeln(cl,' ':margin,'    Type of Wall Structural System (ex: load bearing, veneer, etc.)');
        writeln(cl,' ':margin,'                                                                  ');
        if null(ads row9.rooftype) then row9.rooftype := '                              ';
        if null(ads row9.roofmat) then row9.roofmat := '                              ';
        if null(ads row9.draintyp) then row9.draintyp := '                              ';
        if null(ads row9.draincon) then row9.draincon := '          ';
        writeln(cl,' ':margin,'20. Roof type and drainage system');
```

```
        writeln(cl,' ':margin,'    Roof material: ',row9.roofmat:30);
        writeln(cl,' ':margin,'    Drainage type: ',row9.draintyp:30);
        writeln(cl,' ':margin,'    Drainage condition: ',row9.draincon:10);
        writeln(cl,' ':margin,'  ');
        writeln(cl,' ':margin,'21. Type(s) of Masonry Used in Building');
        if null(ads row10.mastype1) then row10.mastype1 := '                    ';
        if null(ads row10.masqrry1) then row10.masqrry1 := '                    ';
        if null(ads row10.mascour1) then row10.mascour1 := '                    ';
        if null(ads row10.masfin1) then row10.masfin1 := '                    ';
        if null(ads row10.masuse1) then row10.masuse1 := '                    ';
        if null(ads row10.masloc1) then row10.masloc1 := '                    ';
        if null(ads row10.mascond1) then row10.mascond1 := '                    ';
        if null(ads row10.mascolr1) then row10.mascolr1 := '                    ';
        if null(ads row10.mastype2) then row10.mastype2 := '                    ';
        if null(ads row10.masqrry2) then row10.masqrry2 := '                    ';
        if null(ads row10.mascour2) then row10.mascour2 := '                    ';
        if null(ads row10.masfin2) then row10.masfin2 := '                    ';
        if null(ads row10.masuse2) then row10.masuse2 := '                    ';
        if null(ads row10.masloc2) then row10.masloc2 := '                    ';
        if null(ads row10.mascond2) then row10.mascond2 := '                    ';
        if null(ads row10.mascolr2) then row10.mascolr2 := '                    ';
        if null(ads row10.mastype3) then row10.mastype3 := '                    ';
        if null(ads row10.masqrry3) then row10.masqrry3 := '                    ';
        if null(ads row10.mascour3) then row10.mascour3 := '                    ';
        if null(ads row10.masfin3) then row10.masfin3 := '                    ';
        if null(ads row10.masuse3) then row10.masuse3 := '                    ';
        if null(ads row10.masloc3) then row10.masloc3 := '                    ';
        if null(ads row10.mascond3) then row10.mascond3 := '                    ';
        if null(ads row10.mascolr3) then row10.mascolr3 := '                    ';
        writeln(cl,' ':margin-4,'|-------------------------------------------------------------------------------|');
        writeln(cl,' ':margin-4,'|         |   Masonry type #1    |   Masonry type #2    |   Masonry type #3   |');
        writeln(cl,' ':margin-4,'|---------|----------------------|----------------------|---------------------|');
        writeln(cl,' ':margin-4,'|Type     |',row10.mastype1:22,'|',row10.mastype2:22,'|',row10.mastype3:21,'|');
        writeln(cl,' ':margin-4,'|---------|----------------------|----------------------|---------------------|');
        writeln(cl,' ':margin-4,'|Quarry   |',row10.masqrry1:22,'|',row10.masqrry2:22,'|',row10.masqrry3:21,'|');
        writeln(cl,' ':margin-4,'|---------|----------------------|----------------------|---------------------|');
        writeln(cl,' ':margin-4,'|Coursing |',row10.mascour1:22,'|',row10.mascour2:22,'|',row10.mascour3:21,'|');
        writeln(cl,' ':margin-4,'|---------|----------------------|----------------------|---------------------|');
        writeln(cl,' ':margin-4,'|Finish   |',row10.masfin1:22,'|',row10.masfin2:22,'|',row10.masfin3:21,'|');
        writeln(cl,' ':margin-4,'|---------|----------------------|----------------------|---------------------|');
        writeln(cl,' ':margin-4,'|Use      |',row10.masuse1:22,'|',row10.masuse2:22,'|',row10.masuse3:21,'|');
        writeln(cl,' ':margin-4,'|---------|----------------------|----------------------|---------------------|');
        writeln(cl,' ':margin-4,'|Location |',row10.masloc1:22,'|',row10.masloc2:22,'|',row10.masloc3:21,'|');
        writeln(cl,' ':margin-4,'|---------|----------------------|----------------------|---------------------|');
        writeln(cl,' ':margin-4,'|Condition|',row10.mascond1:22,'|',row10.mascond2:22,'|',row10.mascond3:21,'|');
        writeln(cl,' ':margin-4,'|---------|----------------------|----------------------|---------------------|');
        writeln(cl,' ':margin-4,'|Color    |',row10.mascolr1:22,'|',row10.mascolr2:22,'|',row10.mascolr3:21,'|');
        writeln(cl,' ':margin-4,'|---------|----------------------|----------------------|---------------------|');
        writeln(cl,form_feed);
close(cl);
end;

procedure part_one_short;

var     margin  : integer;

begin
        margin := 5;
        rbfind(path[2],relname[2]);
        rbwher(path[2],attlist[1],wherop[1],ads where,wherao[1],natt);
        rbget(path[2],ads row2);
        assign(cl,'lpt1:');
        rewrite(cl);
        writeln(cl);
        writeln(cl,' ':26,'U. S. National Park Service');
```

```
        writeln(cl,' ':margin,' ');
        writeln(cl,' ':margin,'            CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
        writeln(cl);
        writeln(cl,' ':margin,'                              SHORT REPORT     ');
        writeln(cl);
        if null(ads row1.nhl) then row1.nhl := '    ';
        if null(ads row1.fedown) then row1.fedown := '    ';
        if null(ads row1.tra) then row1.tra := '    ';
        writeln(cl,' ':margin,'National Historic Landmark: ',row1.nhl:3,'Federal Property: ',row1.fedown:3,'Tax Act Project: ',
                    row1.tra:3);
        writeln(cl,' ':margin,'-------------------------------------------------------------------------');
        writeln(cl,' ':margin,'PART I  S H O R T   B A C K G R O U N D  I N F O R M A T I O N ');
        writeln(cl,' ':margin,'-------------------------------------------------------------------------');
        writeln(cl,' ':margin,'NAME/LOCATION');
        writeln(cl,' ':margin,'1.  ',row1.bname:40,' (',row1.bnumber:7,')');
        writeln(cl,' ':margin,'   Historical Name of Structure');
        writeln(cl,' ':margin,'   ',row1.altname:40);
        writeln(cl,' ':margin,'   Contemporary Name of Structure (if different)');
        writeln(cl,' ':margin,'    ',row1.bstate:2,'                         ',row1.npsreg:2);
        writeln(cl,' ':margin,'   State                    NPS Region');
        writeln(cl,' ':margin,'   ',row1.bcounty);
        writeln(cl,' ':margin,'   County');
        writeln(cl,' ':margin,'   ',row1.bcity);
        writeln(cl,' ':margin,'   City');
        writeln(cl,' ':margin,'   ',row1.bstreet);
        writeln(cl,' ':margin,'   Street Address (ex. 440 G Street NW)');
        writeln(cl,' ':margin,'   ',row1.bzip);
        writeln(cl,' ':margin,'   Zip Code');
        writeln(cl,' ':margin,'   ',row1.habsno,'       ',row1.butm);
        writeln(cl,' ':margin,'   N.R. number                UTM Coordinates');
        writeln(cl,' ':margin,'              ');
        writeln(cl,' ':margin,'OWNERSHIP/MANAGEMENT');
        writeln(cl,' ':margin,'              ');
        writeln(cl,' ':margin,'   ',row2.oname,'             ', row2.ophone);
        writeln(cl,' ':margin,'    Name of Primary Owner(s)                          Phone #'          );
        writeln(cl,' ':margin);
        rbfind(path[2],relname[3]);
        rbwher(path[2],attlist[1],wherop[1],ads where,wherao[1],natt);
        rbget(path[2],ads row3);
        rbfind(path[2],relname[9]);
        rbwher(path[2],attlist[1],wherop[1],ads where,wherao[1],natt);
        rbget(path[2],ads row9);
        rbfind(path[2],relname[10]);
        rbwher(path[2],attlist[1],wherop[1],ads where,wherao[1],natt);
        rbget(path[2],ads row10);
        writeln(cl,' ':margin,'USE/OCCUPANCY/STATUS' );
        writeln(cl,' ':margin);
        if null(ads row3.nrstat) then row3.nrstat := '    ';
        writeln(cl,' ':margin,'2.  ',row3.nrstat);
        writeln(cl,' ':margin,'    Designation/ National Historic Register');
        writeln(cl,' ':margin);
        writeln(cl,' ':margin,'BUILDING DESCRIPTION');
        writeln(cl,' ':margin);
        if null(ads row9.constdt) then row9.constdt := '        ';
        if null(ads row9.arch) then row9.arch := '                                    ';
        if null(ads row9.builder) then row9.builder := '                                        ';
        writeln(cl,' ':margin,'3.  Dates of construction');
        writeln(cl,' ':margin-4,'-------------------------------------------------------------------------');
        writeln(cl,' ':margin-4,'|  Year  |           Architect            |       Builder/contractor       |');
        writeln(cl,' ':margin-4,'|--------|--------------------------------|--------------------------------|');
        writeln(cl,' ':margin-4,'|',row9.constdt,'|',row9.arch,'|',row9.builder:33,'|');
        writeln(cl,' ':margin-4,'|--------|--------------------------------|--------------------------------|');
        writeln(cl,form_feed);
close(cl);
end;
```

```
procedure part_two_long;

var    margin   : integer;
       s1       : string50;
       s2       : string50;
       s3       : string50;
       s4       : string50;

begin
       s1 := '                                                  ';
       s2 := '                                                  ';
       s3 := '                                                  ';
       s4 := '                                                  ';
        rbget(path[3],ads row11);
        margin := 5;
        assign(cl,'lpt1:');
        rewrite(cl);
        if null(ads row8.mastype) then row8.mastype := '                      ';
        if null(ads row8.masqrry) then row8.masqrry := '                      ';
        if null(ads row8.mascour) then row8.mascour := '                      ';
        if null(ads row8.masfin) then row8.masfin := '                      ';
        if null(ads row8.masuse) then row8.masuse := '                      ';
        if null(ads row8.mascond) then row8.mascond := '                      ';
        if null(ads row8.mascolr) then row8.mascolr := '                      ';
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        if null(ads row8.studyloc) then row8.studyloc := '                          ';
        if null(ads row8.masdens) then row8.masdens := '          ';
        if null(ads row8.masporos) then row8.masporos := '          ';
        if null(ads row8.masstren) then row8.masstren := '          ';
        if null(ads row8.adjmat1) then row8.adjmat1 := '                              ';
        if null(ads row8.adjmat2) then row8.adjmat2 := '                              ';
        if null(ads row8.adjmat3) then row8.adjmat3 := '                              ';
        if null(ads row8.morttype) then row8.morttype := '          ';
        if null(ads row8.mortcolr) then row8.mortcolr :=  '          ';
        if null(ads row8.mortsoft) then row8.mortsoft := '    ';
        if null(ads row8.jointype) then row8.jointype := '          ';
        if null(ads row8.jointdep) then row8.jointdep := '               ';
        if null(ads row8.mortcond) then row8.mortcond := '          ';
        if null(ads row8.mortanal) then row8.mortanal := '    ';
        writeln(cl,' ':margin,'-------------------------------------------------------------------------');
        writeln(cl,' ':margin,'  PART II       M A S O N R Y   S T U D Y   A R E A      ');
        writeln(cl,' ':margin,'-------------------------------------------------------------------------');
        writeln(cl,' ':margin,'22. Location ',row8.studyloc:30,'  (ex: Center, NW side etc.)');
        writeln(cl,' ':margin,'                                                                  ');
        writeln(cl,' ':margin,'                                          24./25. Geological/Physical data' );
        writeln(cl,' ':margin,'23.        Masonry Type                      density ',row8.masdens);
        writeln(cl,' ':margin,'                                             porosity ',row8.masporos);
        writeln(cl,' ':margin,'|----------|---------------------|          strength ',row8.masstren);
        writeln(cl,' ':margin,'|Type      |',row8.mastype:22,'|');
        writeln(cl,' ':margin,'|----------|---------------------|   26. Adjacent Non-masonry Materials');
        writeln(cl,' ':margin,'|Quarry    |',row8.masqrry:22,'|       1) ',row8.adjmat1);
        writeln(cl,' ':margin,'|----------|---------------------|          2) ',row8.adjmat2);
        writeln(cl,' ':margin,'|Coursing  |',row8.mascour:22,'|       3) ',row8.adjmat3);
        writeln(cl,' ':margin,'|----------|---------------------|');
        writeln(cl,' ':margin,'|Finish    |',row8.masfin:22,'|   27. Mortar specifications' );
        writeln(cl,' ':margin,'|----------|---------------------|          Type: ',row8.morttype);
        writeln(cl,' ':margin,'|Use       |',row8.masuse:22,'|       Color: ',row8.mortcolr);
        writeln(cl,' ':margin,'|----------|---------------------|          (H)ard or (S)oft: ',row8.mortsoft);
        writeln(cl,' ':margin,'|Condition |',row8.mascond:22,'|       Joint type: ',row8.jointype);
```

```
        writeln(cl,' ':margin,'|Color     |',row8.mascolr:22,'|       Mortar condition: ',row8.mortcond);
        writeln(cl,' ':margin,'|----------|----------------------|       Mortar analysis (Y/N): ',row8.mortanal);
        writeln(cl,' ':margin,' ');
        writeln(cl,' ':margin,'28. ',row11.untretcd);
        writeln(cl,' ':margin,'    Existing Condition of Untreated Masonry (Supplemented with photographs)');
        writeln(cl,' ':margin,'            ');
        writeln(cl,' ':margin,'29. ',row11.coatcond);
        writeln(cl,' ':margin,'    Existing Condition of Surface Coating  (if applicable)');
        writeln(cl,' ':margin,'   ');
        writeln(cl,' ':margin,'30. ',row11.moistprb,' 32. ',row11.specs);
        writeln(cl,' ':margin,'    Evidence of Moisture Problems (Supplement with photographs)   Specs Y/N');
        writeln(cl,' ':margin,'      ');
        writeln(cl,' ':margin,'31. Previous Treatment      Date        Previous Treatment         Date');
        writeln(cl,' ':margin-2,row11.trt1,' ',row11.trtdate1,' ',row11.trt3,' ',row11.trtdate3);
        writeln(cl,' ':margin-2,row11.trt2,' ',row11.trtdate2,' ',row11.trt4,' ',row11.trtdate4);
        writeln(cl,' ':margin,'        ');
        writeln(cl,' ':margin,'33. Describe Masonry Condition as a Result of Previous Treatment');
        bstring(row11.precond,s1,s2,s3,s4);
        writeln(cl,' ':margin,'        ',s1);
        writeln(cl,' ':margin,'        ',s2);
        writeln(cl,' ':margin,'        ',s3);
        writeln(cl,' ':margin,'        ',s4);
        writeln(cl,' ':margin,'34. Describe Condition of Masonry requiring treatment prescibed in Part III');
       s1 := '                                                ';
       s2 := '                                                ';
       s3 := '                                                ';
       s4 := '                                                ';
        bstring(row11.detdes,s1,s2,s3,s4);
        writeln(cl,' ':margin,'        ',s1);
        writeln(cl,' ':margin,'        ',s2);
        writeln(cl,' ':margin,'        ',s3);
        writeln(cl,' ':margin,'        ',s4);
       s1 := '                                                ';
       s2 := '                                                ';
       s3 := '                                                ';
       s4 := '                                                ';
        writeln(cl,' ':margin,' ');
        writeln(cl,' ':margin,' ');
        writeln(cl,' ':margin+50,'Masonry type number: ',row11.number:2);
        writeln(cl,form_feed);
        close(cl);
end;

procedure part_two_short;

var    margin   : integer;

begin
        margin := 5;
        assign(cl,'lpt1:');
        rewrite(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl,' ':margin,'-------------------------------------------------------------------------------');
        writeln(cl,' ':margin,'  PART II     S H O R T    M A S O N R Y   S T U D Y   A R E A      ');
        writeln(cl,' ':margin,'-------------------------------------------------------------------------------');
        writeln(cl);
        writeln(cl);
        if null(ads row8.mastype) then row8.mastype := '                         ';
        if null(ads row8.masqrry) then row8.masqrry := '                         ';
```

```
        if null(ads row8.masfin) then row8.masfin := '                          ';
        if null(ads row8.masuse) then row8.masuse := '                          ';
        if null(ads row8.mascond) then row8.mascond := '                          ';
        if null(ads row8.mascolr) then row8.mascolr := '                          ';
        writeln(cl,' ':margin,' 4.                              Masonry type        ' );
        writeln(cl,' ':margin+20,'|----------|----------------------| ');
        writeln(cl,' ':margin+20,'|Type      |',row8.mastype:22,'|');
        writeln(cl,' ':margin+20,'|----------|----------------------| ');
        writeln(cl,' ':margin+20,'|Quarry    |',row8.masqrry:22,'|   ');
        writeln(cl,' ':margin+20,'|----------|----------------------|');
        writeln(cl,' ':margin+20,'|Coursing  |',row8.mascour:22,'|   ');
        writeln(cl,' ':margin+20,'|----------|----------------------|');
        writeln(cl,' ':margin+20,'|Finish    |',row8.masfin:22,'|  ' );
        writeln(cl,' ':margin+20,'|----------|----------------------|');
        writeln(cl,' ':margin+20,'|Use       |',row8.masuse:22,'|    ');
        writeln(cl,' ':margin+20,'|----------|----------------------|');
        writeln(cl,' ':margin+20,'|Condition |',row8.mascond:22,'|   ');
        writeln(cl,' ':margin+20,'|----------|----------------------|');
        writeln(cl,' ':margin+20,'|Color     |',row8.mascolr:22,'|   ');
        writeln(cl,' ':margin+20,'|----------|----------------------|');
        writeln(cl,' ':margin,' ');
        close(cl);

end;


procedure part_three_long;

var     margin    : integer;
        s1,s2,s3,s4 : string50;

begin
        assign(cl,'lpt1:');
        rewrite(cl);
        margin := 5;
        if row7.number = row8.number then begin
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl,' ':margin,'-------------------------------------------------------------------------------');
        writeln(cl,' ':margin,'PART III     T R E A T M E N T S    B E I N G   M O N I T O R E D');
        writeln(cl,' ':margin,'-------------------------------------------------------------------------------');
        writeln(cl,' ':margin,row7.trttype:65);
        writeln(cl,' ':margin,'                                                  ',row7.begdate,'       ',row7.enddate);
        writeln(cl,' ':margin,'35. Treatment to be Monitored                    Start Date      End Date');
        writeln(cl,' ':margin,'36. Was Treatment Tested on Building Before Application (Y/N) ? ',row7.tested:1);
        writeln(cl,' ':margin,'    If so; when and for how long; and where on building ---' );
        bstring(row7.testdes,s1,s2,s3,s4);
        writeln(cl,' ':margin,'          ',s1:50);
        writeln(cl,' ':margin,'          ',s2:50);
        writeln(cl,' ':margin,'          ',s3:50);
        writeln(cl,' ':margin,'          ',s4:50);
        writeln(cl,' ':margin,'                                                                  ');
       s1 := '                                          ';
       s2 := '                                          ';
       s3 := '                                          ';
       s4 := '                                          ';
        if null(ads row7.prod1) then row7.prod1 := '                                                                  ';
        if null(ads row7.prod2) then row7.prod2 := '                                                                  ';
        if null(ads row7.prod3) then row7.prod3 := '                                                                  ';
        writeln(cl,' ':margin,'37. Name of Product(s) used if Applicable');
        writeln(cl,' ':margin,'    1) ',row7.prod1);
```

```
        writeln(cl,' ':margin,'    3) ',row7.prod3);
        writeln(cl,' ':margin,'           ');
        writeln(cl,' ':margin,'38. Temperature ',row7.temp:10,'    Humidity ',row7.humidity:10,'   Precipitation ',row7.precip:10)
        writeln(cl,' ':margin,'    Average Weather Conditions at Time of Treatment');
        writeln(cl,' ':margin,'39. ',row7.areatrt);
        writeln(cl,' ':margin,'    Location of Treatment (define a limited area)' );
        writeln(cl,' ':margin,'       ');
        writeln(cl,' ':margin,'40. Describe Immediate Post-Treatment Appearance of Masonry');
        bstring(row7.appear,s1,s2,s3,s4);
        writeln(cl,' ':margin,'           ',s1:50);
        writeln(cl,' ':margin,'           ',s2:50);
        writeln(cl,' ':margin,'           ',s3:50);
        writeln(cl,' ':margin,'           ',s4:50);
       s1 := '                                                ';
       s2 := '                                                ';
       s3 := '                                                ';
       s4 := '                                                ';
        writeln(cl,' ':margin+50,'Masonry type number: ',row7.number:2);
        writeln(cl,' ':margin+50,'Treatment number   : ',row7.tnum:2);
        writeln(cl,form_feed);
        end;
        close(cl);
end;

procedure part_three_short;

var     margin    : integer;
        s1,s2,s3,s4 : string50;

begin
        assign(cl,'lpt1:');
        rewrite(cl);
        margin := 5;
        if row7.number = row8.number then begin
        if (n mod 2) = 0 then begin
        writeln(cl,form_feed);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        end;
        n := n + 1;
        if null(ads row7.begdate) then row7.begdate := '        ';
        if null(ads row7.enddate) then row7.enddate := '        ';
        if null(ads row7.trttype)
           then
           row7.trttype := '                                                                                ';
        writeln(cl,' ':margin,'-------------------------------------------------------------------------------');
        writeln(cl,' ':margin,'PART III                 S H O R T   T R E A T M E N T S  ');
        writeln(cl,' ':margin,'-------------------------------------------------------------------------------');
        writeln(cl,' ':margin,' ');
        writeln(cl,' ':margin,'5.  ',row7.begdate,'        ',row7.enddate);
        writeln(cl,' ':margin,'     Start Date        End Date');
        writeln(cl,' ':margin,' ');
        writeln(cl,' ':margin,'6. ',row7.trttype:60);
        writeln(cl,' ':margin,'     ');
        writeln(cl,' ':margin,'    Treatment to be Monitored ');
        writeln(cl,' ':margin,'     ');
        writeln(cl,' ':margin,'    Description of the test procedures' );
        bstring(row7.testdes,s1,s2,s3,s4);
        writeln(cl,' ':margin,'           ',s1:50);
        writeln(cl,' ':margin,'           ',s2:50);
        writeln(cl,' ':margin,'           ',s3:50);
        writeln(cl,' ':margin,'           ',s4:50);
```

```
      s1 := '                                                  ';
      s2 := '                                                  ';
      s3 := '                                                  ';
      s4 := '                                                  ';
       if null(ads row7.prod1) then row7.prod1 := '                                                                    ';
       if null(ads row7.prod2) then row7.prod2 := '                                                                    ';
       if null(ads row7.prod3) then row7.prod3 := '                                                                    ';
       writeln(cl,' ':margin,'7.  Name of Product(s) used if Applicable');
       writeln(cl,' ':margin,'    1) ',row7.prod1);
       writeln(cl,' ':margin,'    2) ',row7.prod2);
       writeln(cl,' ':margin,'    3) ',row7.prod3);
       writeln(cl,' ':margin,'            ');
       writeln(cl,' ':margin,'    ',row7.areatrt);
       writeln(cl,' ':margin,'8.  Extent of treatment ( Area treated): ');
       writeln(cl,' ':margin,'  ');
       writeln(cl,' ':margin,'9. Reason for the treatment ');
       bstring(row7.appear,s1,s2,s3,s4);
       writeln(cl,' ':margin,'           ',s1:50);
       writeln(cl,' ':margin,'           ',s2:50);
       writeln(cl,' ':margin,'           ',s3:50);
       writeln(cl,' ':margin,'           ',s4:50);
       writeln(cl,' ':margin,'      ');
      s1 := '                                                  ';
      s2 := '                                                  ';
      s3 := '                                                  ';
      s4 := '                                                  ';
       writeln(cl,' ':margin+50,'Masonry type number: ',row7.number:2);
       writeln(cl,' ':margin+50,'Treatment number   : ',row7.tnum:2);
       if (n mod 2) <> 0 then writeln(cl,form_feed);
       end;
       close(cl);
end;


procedure part_four_long;

begin
       margin := 5;
       assign(cl,'lpt1:');
       rewrite(cl);
       writeln(cl);
       writeln(cl);
       writeln(cl);
       writeln(cl);
       writeln(cl);
       writeln(cl);
       writeln(cl,' ':margin,'RECORDERS INFORMATION');
       writeln(cl);
       writeln(cl,' ':margin,'NAME & ADDRESS OF RECORDER');
       writeln(cl,' ':margin,' ');
       writeln(cl,' ':margin,'42. ',row4.rname:40,'  ',row4.rphone:14);
       writeln(cl,' ':margin,'    Name (Last name first)                  Phone #');
       writeln(cl,' ':margin,'    Title');
       writeln(cl,' ':margin,'  ',row4.rstreet:30,'    ',row4.rorgan:30);
       writeln(cl,' ':margin,'    Address                                Organization');
       writeln(cl,' ':margin,'  ',row4.rcity:20,'       ',row4.rstate:2,'          ',row4.rzip:10);
       writeln(cl,' ':margin,'    City                    State          Zip');
       writeln(cl,' ':margin,'  ');
       writeln(cl,' ':margin,'  ');
       writeln(cl,' ':margin,'    Date of Entry');
       writeln(cl,' ':margin,'    ',row4.entdate:8);
       writeln(cl,' ':margin,'                  ------> Refer any questions to <----------');
       writeln(cl,' ':margin,'                  | Preservation Assistance Division (424) |');
       writeln(cl,' ':margin,'                  |         National Park Service          |');
       writeln(cl,' ':margin,'                  |           P.O. Box  37127              |');
       writeln(cl,' ':margin,'                  |      Washington, D.C. 20013-7127       |');
```

```
        writeln(cl,' ':margin,'                    ----------------------------------------');
        writeln(cl,form_feed);
        close(cl);
end;

procedure part_four_short;


begin
        margin := 5;
        assign(cl,'lpt1:');
        rewrite(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl,' ':margin,'RECORDERS INFORMATION');
        writeln(cl);
        writeln(cl,' ':margin,'NAME & ADDRESS OF RECORDER');
        writeln(cl,' ':margin,' ');
        writeln(cl,' ':margin,'10. ',row4.rname,'  ',row4.rphone);
        writeln(cl,' ':margin,'    Name (Last name first)                    Phone #');
        writeln(cl,' ':margin,'   ');
        writeln(cl,' ':margin,'   ');
        writeln(cl,' ':margin,'   ');
        writeln(cl,' ':margin,'   ');
        writeln(cl,' ':margin,'   ');
        writeln(cl,' ':margin,'   ');
        writeln(cl,' ':margin,'   ');
        writeln(cl,' ':margin,'   ');
        writeln(cl,' ':margin,'   ');
        writeln(cl,' ':margin,'   ');
        writeln(cl,' ':margin,'   ');
        writeln(cl,' ':margin,'   ');
        writeln(cl,' ':margin,'   ');
        writeln(cl,' ':margin,'   ');
        writeln(cl,' ':margin,'   ');
        writeln(cl,' ':margin,'   ');
        writeln(cl,' ':margin,'    Date of Entry');
        writeln(cl,' ':margin,'    ',row4.entdate);
        writeln(cl,' ':margin,'                    ------> Refer any questions to <----------');
        writeln(cl,' ':margin,'                    | Preservation Assistance Division (424) |');
        writeln(cl,' ':margin,'                    |         National Park Service          |');
        writeln(cl,' ':margin,'                    |           P.O. Box  37127              |');
        writeln(cl,' ':margin,'                    |      Washington, D.C. 20013-7127       |');
        writeln(cl,' ':margin,'                    |             (202)343-9567              |');
        writeln(cl,' ':margin,'                    ----------------------------------------');
        writeln(cl,form_feed);
        close(cl);
end;

procedure print_follow_up;

var     margin    : integer;
        s1,s2,s3,s4 : string50;

begin
        margin := 5;
        assign(cl,'lpt1:');
    attlist[2] := 'ENTDATE ';
    wherop[2] := 'EQ      ';
    wherao[2] := 'AND ';
    natt := 2;
      where.int[6] := 8;
```

```
rbfind(path[3],relname[12]);
rbcheck('rbfind  ',3);
rbwher(path[3],attlist[1],wherop[1], ads where,wherao[1],natt);
rbcheck('rbwher  ',3);
rewrite(cl);
writeln(cl);
writeln(cl,'                              U. S. National Park Service');
writeln(cl);
writeln(cl);
writeln(cl,'                         CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
writeln(cl);
writeln(cl,'                                   FOLLOW-UP REPORT ');
writeln(cl);
writeln(cl,' ':margin,'-------------------------------------------------------------------------');
writeln(cl,' ':margin,'      F O L L O W - U P   I N S P E C T I O N   I N F O R M A T I O N ');
writeln(cl,' ':margin,'-------------------------------------------------------------------------');
writeln(cl,' ':margin);
writeln(cl,' ':margin,'    Entry Date: ',row6.entdate:8);
writeln(cl,' ':margin,'     ');
writeln(cl,' ':margin,'1.  ',row6.bname:40);
writeln(cl,' ':margin,'    Name of Structure');
writeln(cl,' ':margin);
writeln(cl,' ':margin,'2.  ',row6.oname:40,'                  ',row6.ophone:14);
writeln(cl,' ':margin,'    Name of Primary Owner(s)                              Phone #');
writeln(cl,' ':margin);
writeln(cl,' ':margin,'    ',row6.ostreet:30);
writeln(cl,' ':margin,'    Street Address');
writeln(cl,' ':margin);
writeln(cl,' ':margin,'    ',row6.ocity:20,'       ',row6.ostate:2,'            ',row6.ozip:10);
writeln(cl,' ':margin,'    City                    State                         Zip');
writeln(cl,' ':margin);
n := 0;
repeat
    rbget(path[3],ads row12);
    rbstat(rbs2);
    if rbs2 <> -1 then begin
    n := n + 1;
    if (n mod 2) = 0 then begin
        writeln(cl,form_feed);
        writeln(cl);
        writeln(cl);
writeln(cl,'                              U. S. National Park Service');
writeln(cl);
writeln(cl);
writeln(cl,'                         CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
writeln(cl);
writeln(cl,'                                   FOLLOW-UP REPORT ');
        writeln(cl);
        writeln(cl);
        writeln(cl,' ':margin,'-------------------------------------------------------------------------');
        writeln(cl,' ':margin,'      F O L L O W - U P   I N S P E C T I O N   I N F O R M A T I O N ');
        writeln(cl,' ':margin,'-------------------------------------------------------------------------');
        end;
writeln(cl,' ':margin,row12.trttype:70);
writeln(cl,' ':margin,'3. Treatment being monitored');
writeln(cl,' ':margin,'    ',row12.mastype:24,'      ',row12.trtloc:30,'  ',row12.trtdate:8);
writeln(cl,' ':margin,'     Masonry Material             Location on building    Date');
writeln(cl);
if null(ads row12.prod1) then row12.prod1 := '                                                                    ';
if null(ads row12.prod2) then row12.prod2 := '                                                                    ';
writeln(cl,' ':margin,'4. Name of Product(s) used if Applicable');
writeln(cl,' ':margin,'a) ',row12.prod1:60);
writeln(cl,' ':margin,'b) ',row12.prod2:60);
writeln(cl);
writeln(cl,' ':margin,'5. Has treatment changed appearance of area? ',row12.matchung:1,'  Describe below:');
```

```
writeln(cl,' ':margin,'           ',s1:50);
writeln(cl,' ':margin,'           ',s2:50);
writeln(cl,' ':margin,'           ',s3:50);
writeln(cl,' ':margin,'           ',s4:50);
writeln(cl);
s1 := '                                                  ';
s2 := '                                                  ';
s3 := '                                                  ';
s4 := '                                                  ';
writeln(cl,' ':margin,'6. Visual description of treated area');
bstring(row12.areades,s1,s2,s3,s4);
writeln(cl,' ':margin,'           ',s1:50);
writeln(cl,' ':margin,'           ',s2:50);
writeln(cl,' ':margin,'           ',s3:50);
writeln(cl,' ':margin,'           ',s4:50);
writeln(cl,' ':margin,'      ');
s1 := '                                                  ';
s2 := '                                                  ';
s3 := '                                                  ';
s4 := '                                                  ';
writeln(cl,' ':margin,'7. Is another treatment type planned? ',row12.futtrt:1,'   Explain below:');
bstring(row12.futdes,s1,s2,s3,s4);
writeln(cl,' ':margin,'           ',s1:50);
writeln(cl,' ':margin,'           ',s2:50);
writeln(cl,' ':margin,'           ',s3:50);
writeln(cl,' ':margin,'           ',s4:50);
writeln(cl,' ':margin,'      ');
s1 := '                                                  ';
s2 := '                                                  ';
s3 := '                                                  ';
s4 := '                                                  ';
writeln(cl,' ':margin,'8. Time elapsed: ',row12.lapsetm:10);
writeln(cl,' ':margin,'9. Will this treatment be repeated? ',row12.reptrt:1);
writeln(cl,' ':margin,'   If so when',row12.repdate:8);
writeln(cl,' ':margin,'      ');
writeln(cl,' ':margin,'10. Comments:');
bstring(row12.comments,s1,s2,s3,s4);
writeln(cl,' ':margin,'           ',s1:50);
writeln(cl,' ':margin,'           ',s2:50);
writeln(cl,' ':margin,'           ',s3:50);
writeln(cl,' ':margin,'           ',s4:50);
writeln(cl,' ':margin,'      ');
s1 := '                                                  ';
s2 := '                                                  ';
s3 := '                                                  ';
s4 := '                                                  ';
    end;
until rbs2 = -1;
rbfind(path[4],relname[13]);
rbcheck('rbfind  ',4);
rbwher(path[4],attlist[1],wherop[1], ads where,wherao[1],natt);
rbcheck('rbwher  ',4);
rbget(path[4],ads row13);
    if (n mod 2) <> 0 then begin
        writeln(cl,form_feed);
        writeln(cl);
        writeln(cl);
writeln(cl,'                              U. S. National Park Service');
writeln(cl);
writeln(cl);
writeln(cl,'                         CENSUS OF TREATED HISTORIC MASONRY BUILDINGS');
writeln(cl);
writeln(cl,'                                    FOLLOW-UP REPORT ');
        writeln(cl);
        writeln(cl);
```

```
            writeln(cl,' ':margin,'     F O L L O W - U P   I N S P E C T I O N   I N F O R M A T I O N ');
            writeln(cl,' ':margin,'-------------------------------------------------------------------------------');
            end;
            writeln(cl);
        writeln(cl,' ':margin,'11. NAME & ADDRESS OF RECORDER ');
        writeln(cl,' ':margin,'   ');
        writeln(cl,' ':margin,'    ',row13.rname:40,'     ',row13.rphone:14);
        writeln(cl,' ':margin,'    Name (Last name first)                  Phone #');
        writeln(cl,' ':margin,'    ',row13.rtitle:30);
        writeln(cl,' ':margin,'    Title');
        writeln(cl,' ':margin,'    ',row13.rstreet:30,'       ',row13.rorgan:30);
        writeln(cl,' ':margin,'    Address                                Organization');
        writeln(cl,' ':margin,'    ',row13.rcity:20,'      ',row13.rstate:2,'           ',row13.rzip:10);
        writeln(cl,' ':margin,'    City                   State          Zip');
        writeln(cl);
        writeln(cl);
        writeln(cl);
        writeln(cl,' ':margin,'                  ------> Refer any questions to <----------');
        writeln(cl,' ':margin,'                  | Preservation Assistance Division (424) |');
        writeln(cl,' ':margin,'                  |        National Park Service           |');
        writeln(cl,' ':margin,'                  |          P.O. Box  37127               |');
        writeln(cl,' ':margin,'                  |     Washington, D.C. 20013-7127        |');
        writeln(cl,' ':margin,'                  |            (202)343-9567               |');
        writeln(cl,' ':margin,'                  ----------------------------------------');
        writeln(cl,form_feed);
        close(cl);
end;

procedure print_follow;

begin
     attlist[1] := 'BNUMBER ';
     wherop[1] := 'EQ      ';
     wherao[1] := 'AND ';
     natt := 1;
     b_ask;
     clear_screen(20,0,24,79);
        rbfind(path[2],relname[6]);
        rbcheck('rbfind  ',2);
        rbwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
        rbstat(rbs);
        if rbs <> -1 then begin
        repeat
           natt := 1;
             bright;cur_pos(22,20);write('Prepare Printer for Follow-up Report');
             cur_pos(23,25);write('Press any key to continue ');
             spchin(ch);
           rbget(path[2],ads row6);
           rbstat(rbs1);
           if rbs1 <> -1 then print_follow_up;
        until rbs1 = -1;
        end else begin
           cur_pos(19,27);write('No FOLLOW-UP reports exist for this building');
        end;
 end;



procedure comp;

begin
     attlist[1] := 'BNUMBER ';
     wherop[1] := 'EQ      ';
     wherao[1] := 'AND ';
     natt := 1;
```

```
    clear_screen(20,0,24,79);
    bright;cur_pos(22,20);write('Prepare Printer for Comprehensive Report');
    cur_pos(23,25);write('Press any key to continue ');
    spchin(ch);
       rbfind(path[1],relname[1]);
       rbcheck('rbfind  ',1);
       rbwher(path[1],attlist[1],wherop[1], ads where,wherao[1],natt);
       rbcheck('rbwher  ',1);
       rbget(path[1],ads row1);
       if bnum[3] = 'C' then part_one_long else part_one_short;
       rbfind(path[2],relname[8]);
       rbcheck('rbfind  ',2);
       rbwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
       rbcheck('rbwher  ',2);
       rbfind(path[3],relname[11]);
       rbcheck('rbfind  ',3);
       rbwher(path[3],attlist[1],wherop[1], ads where,wherao[1],natt);
       rbcheck('rbwher  ',3);
         repeat
           natt := 1;
           rbget(path[2],ads row8);
           rbstat(rbs1);
           if rbs1 <> -1 then begin
           if bnum[3] = 'C' then part_two_long else part_two_short;
           natt := 1;
           rbfind(path[4],relname[7]);
           rbcheck('rbfind  ',4);
           rbwher(path[4],attlist[1],wherop[1], ads where,wherao[1],natt);
           rbcheck('rbwher  ',4);
           n := 0;
           rbs2 := 0;
           repeat
             rbget(path[4],ads row7);
             rbstat(rbs2);
             if rbs2 <> -1 then begin
             if bnum[3] = 'C' then part_three_long else part_three_short;
           end;
           until rbs2 = -1;
         end;
         until rbs1 = -1;
       attlist[1] := 'BNUMBER ';
       wherop[1] := 'EQ      ';
       wherao[1] := 'AND ';
       natt := 1;
       rbfind(path[2],relname[4]);
       rbwher(path[2],attlist[1],wherop[1], ads where,wherao[1],natt);
       rbget(path[2],ads row4);
       if bnum[3] = 'C' then part_four_long else part_four_short;
end;



end.
```

```
{$include:'c.pas'}

program CEN(input,output);

uses census;

procedure print_follow;extern;
procedure comp; extern;


{$include:'csform.pas'}
{$include:'cadd.pas'}
{$include:'cform.pas'}
{$include:'cmenu.pas'}

VAR     dwayne_null : names;
        space       : char;

begin
        dbname := 'C:CENSUS ';
        rbstrt;
        blank := ' ';
        dataerr := false;
        edfunct := 0;
        dwayne_null := 'NULL    ';
        space := chr(32);
        rbopen (dbname);
        rbset(dwayne_null,space);
        path[1] := 1 ;
        path[2] := 2 ;
        path[3] := 3 ;
        path[4] := 4 ;
        path[5] := 5 ;
        relname[1] := 'IDENT1  ';
        relname[2] := 'IDENT2  ';
        relname[3] := 'IDENT3  ';
        relname[4] := 'IDENT4  ';
        relname[5] := 'SHORTID ';
        relname[6] := 'FOLLOW1 ';
        relname[7] := 'TREATMNT';
        relname[8] := 'STUDY1  ';
        relname[9] := 'MAT1    ';
        relname[10] := 'MAT2    ';
        relname[11] := 'STUDY2  ';
        relname[12] := 'FOLLOW2 ';
        relname[13] := 'FOLLOW3 ';
        date(todays_date);
        time_mark;
        scrint;
        caps_lock;
        clear_screen(2,0,23,79);
main_menu;

end.
```